UNISIM
Virtex 5 FXT Simulator Manual

Gilles Mouchard

Contents

1 Introduction **4**
1.1 UNISIM . . . 4
1.2 Virtex 5 FXT Simulator . . . 4

2 Building the simulator **7**
2.1 Requirements . . . 7
2.2 Installing SystemC 2.2.0 and SystemC TLM 2.0.1 . . . 7
2.2.1 Download the source code . . . 7
2.2.2 Uncompressing the source code tarballs . . . 7
2.2.3 Patching the source code (recommended for g++ ≥ 4.1) . . . 8
2.2.4 Configuring SystemC . . . 8
2.2.5 Compiling and installing SystemC . . . 8
2.3 Building the UNISIM Virtex 5 FXT simulator . . . 8
2.3.1 Uncompressing the source code tarball . . . 8
2.3.2 Configuring the simulator building process . . . 8
2.3.3 Compiling the simulator . . . 9

3 Cross-compiling the simulator **9**
3.1 Requirements . . . 9
3.2 Installing a cross-compiled SystemC 2.2.0 and SystemC TLM 2.0.1 . . . 9
3.2.1 Download the source code . . . 9
3.2.2 Uncompressing the source code tarballs . . . 9
3.2.3 Patching the source code (recommended for g++ ≥ 4.1) . . . 10
3.2.4 Configuring SystemC . . . 10
3.2.5 Cross-compiling and installing SystemC . . . 10
3.3 Cross-compiling zlib . . . 10
3.4 Cross-compiling libxml2 . . . 10
3.5 Cross-compiling Boost . . . 11
3.6 Cross-compiling the UNISIM Virtex 5 FXT simulator . . . 11
3.6.1 Uncompressing the source code tarball . . . 11
3.6.2 Configuring the simulator building process . . . 11
3.6.3 Cross-compiling the simulator . . . 11

4 Getting started **12**
4.1 Run-time configuration . . . 12
4.2 Loading binaries . . . 12
4.3 Serial console . . . 13
4.4 Using the builtin debugger . . . 13
4.5 Using a GNU crosstool chain . . . 13
4.5.1 Building a GNU crosstool chain . . . 13

4.5.2 Using cross-GCC . . . 16
4.5.3 Using cross-GDB . . . 16

5 Examples of use **16**
5.1 Building the examples . . . 16
5.1.1 Building the basic system level unit tests . . . 16
5.1.2 Building the MiBench version 1 benchmarks . . . 17
5.1.3 Building the light weight Linux distribution . . . 17
5.2 Basic system level tests . . . 17
5.2.1 Echo . . . 19
5.2.2 Poll . . . 19
5.2.3 Gen . . . 20
5.2.4 Cap . . . 21
5.2.5 PWM . . . 22
5.2.6 Autoselect . . . 23
5.2.7 CFI query . . . 23
5.2.8 Chip erase . . . 24
5.2.9 Sector erase . . . 24
5.2.10 Single word programming . . . 25
5.2.11 Unlock bypass chip erase . . . 25
5.2.12 Unlock bypass sector erase . . . 26
5.2.13 Unlock bypass word programming . . . 26
5.2.14 Write buffer programming . . . 27
5.3 A light weight Linux distribution . . . 28
5.3.1 The boot program . . . 28
5.3.2 The Linux kernel . . . 28
5.3.3 The device Tree . . . 30
5.3.4 The initial RAM disk . . . 30
5.3.5 Busybox . . . 33
5.3.6 The MiBench version 1 benchmarks . . . 33
5.3.7 Running the example . . . 33

A Simulator technical reference (generated) **37**
A.1 Introduction . . . 37
A.2 Licensing . . . 37
A.3 Simulated configuration . . . 37
A.4 Using the UNISIM Virtex 5 FXT simulator . . . 42
A.5 Configuration . . . 42
A.6 Statistics . . . 106
A.7 Formulas . . . 110

List of Figures

1 UNISIM Virtex 5 FXT simulator schematic. . . . 5
2 crosstool chain build configuration (changes from default only) for Crosstool-NG 1.11.3. . . . 15
3 Boot program (boot.S compiled as boot.elf) loaded in BRAM. . . . 29
4 Linux kernel configuration (changes from default only). . . . 31
5 UNISIM Virtex 5 FXT simulator schematic. . . . 37

List of Tables

1 Simulator memory mapping. 5
2 Summary of basic system level tests. 18
3 Files for booting Linux in the simulator. 28
4 Linux kernel register parameters. 29
5 Initial RAM disks. 32
6 MiBench version 1.0. 34

1 Introduction

1.1 UNISIM

UNISIM provides several virtual platforms and a framework to ease the development of new virtual platforms. A virtual platform is a software tool, often called simulator, that mimics the behavior of an electronic system so that software can run on it before silicon or FPGA implementation of that electronic system is available. The simulated electronic system can include lots of microprocessors and devices. Depending on the needed representativeness and simulator development budget, a simulator can be as simple as an instruction set simulator as well as a full system simulator. A full system simulator, not only executes the microprocessor instruction set, like an instruction set simulator, but also simulates buses, I/O devices, sensors, actuators, so that real application workloads and operating systems can run on them. Most of UNISIM virtual platforms are full system simulators, which means that they are sufficiently representative of the real hardware that whole operating systems (e.g. Linux, VxWorks), unmodified software stacks (e.g. an AUTOSAR software stack), and industrial applications can run on them. The UNISIM virtual platforms are modular: a simulator is the assembly of properly configured simulation components (e.g. CPU, RAM, buses). They are written in C/C++ and based on industry standards, like IEEE1666^{TM}, OSCI SystemC^{TM} and OSCI SystemC^{TM} TLM 2.0.
Some use cases of UNISIM virtual platforms are:

- Development of SystemC IPs (intellectual property) and new virtual platforms: UNISIM is an open development environment that comprise a SystemC module library, and a set of services (debugging, program loaders,). It can be a fundation for the development of new SystemC IPs and new virtual virtual platforms.

- Hybrid virtual platform: UNISIM/SystemC and an FPGA accelerator can be mixed to build some hybrid virtual platforms: for instance simulating processor cores within UNISIM/SystemC, and prototyping specialized IPs/devices within an FPGA accelerator. Hybridization allows using indifferently both UNISIM/SystemC IPs (on a standard host machine) and VHDL IPs (on an FPGA accelerator), but also speeding up simulation of large systems.

- Non-intrusive debugging and testing of software: It means that, unlike on the real hardware, software can be debugged and tested without affecting either its functional and/or temporal behavior. With such virtual instrumentation, the user can seamlessly stop and resume execution of software, profile the software, inspect the system status, inject values on the sensors, modify the state of program variables and microprocessor/device registers, and then analyze the result without modifying the software.

- Hardware/software integration: software stack can be debugged and tested within a representative hardware environment before the availability of either the FPGA prototypes or the real hardware. The software stack can be composed of low level software (e.g. drivers), of a real-time operating system, and of applications generated from high level models (Papyrus, Matlab Simulink, Statemate Stateflow,)

Several open source virtual platforms for different targets (ARM, PowerPC, Star12X, and TMS320C3X) and different hosts (Linux, Windows, Mac OS X) are available for download here. These virtual platforms have been evaluated and used in various industry domains such as automotive, avionic, military, electrical equipments for medium tension, nuclear safety.

1.2 Virtex 5 FXT Simulator

The UNISIM Virtex 5 FXT is a virtual platform that tries to mimic a Xilinx ML507 development board that include a Xilinx Virtex 5 FXT (FPGA + PPC440).

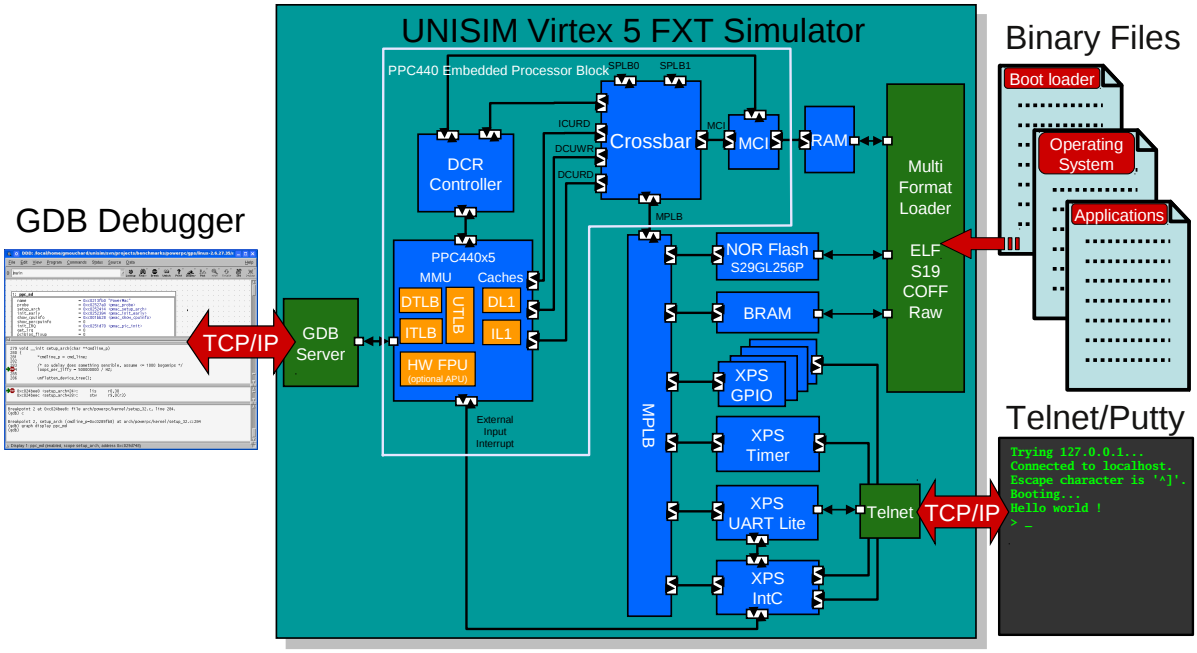

Figure 1: UNISIM Virtex 5 FXT simulator schematic.

Address range	Component name	Hardware device
0x00000000-0x0003ffff	ram	256 MB RAM
0x81400000-0x8140ffff	gpio-leds-8bit	XPS GPIO
0x81420000-0x8142ffff	gpio-5-leds-positions	XPS GPIO
0x81440000-0x8144ffff	gpio-push-buttons-5bit	XPS GPIO
0x81460000-0x8146ffff	gpio-dip-switches-8bit	XPS GPIO
0x81800000-0x8180ffff	intc	XPS IntC
0x83c00000-0x83c0ffff	timer	XPS Timer/Counter
0x84000000-0x8400ffff	uart-lite	XPS UART Lite
0xfc000000-0xfdffffff	flash	S29GL256P
0xfffc0000-0xffffffff	bram	256 KB BRAM

Table 1: Simulator memory mapping.

The UNISIM Virtex 5 FXT simulator, which Figure 1 shows the schematic, implements the following:

- PPC440 Embedded Processor Block (UG200 [9]):
  - All the PPC440x5 [4] instruction set of a Xilinx Virtex 5 FXT
  - Optional FPU that is similar to Xilinx FPU APU [12]
  - PowerPC Book E MMU (shadow ITLB, shadow DTLB, unified TLB)
  - Caches (instruction and data)
  - Integrated timers (decrementer, fixed interval, watchdog)
  - Exception handling mechanisms
  - DCR (device control register) bus controller
  - Crossbar
  - MCI (Memory Controller Interface)
  - MPLB (master processor local bus) interface
  - SPLB0 and SPLB1 (slave processor local bus) interfaces
- MPLB (master processor local bus) link
- 256 MB RAM on MCI
- 256 KB BRAM on MPLB
- XPS IntC interrupt controller (v2.01a) [10] on MPLB
- XPS Timer/Counter (v1.02a) [11] on MPLB
- Spansion 256 Mbits (32 MB) S29GL256P off-chip flash memory [6] on MPLB
- XPS UART Lite (v1.01a) [8] on MPLB
- Four XPS GPIO modules (v2.00a) [7] on MPLB connected to two LED boards and two DIP switch/push buttons boards

Several stub modules are currently integrated in the simulator to test the XPS Timer/Counter module:

- GenerateOut stubs connected on XPS Timer/Counter GenerateOut outputs
- PWM stub connected on XPS Timer/Counter PWM0 output
- CaptureTrigger stubs (optional randomized outputs) connected on XPS Timer/Counter CaptureTrigger inputs
- SPLB stubs
- DCR stubs

The simulator also supports the following features:

- Loading of ELF32, ELF64 [3] and Motorola S19 (S-Record) [2] files
- An integrated console debugger that supports debugging both at assembly level and source level (e.g. C source code). Source level debugging is only available for ELF binary files including DWARF v2 or v3 [5] debugging informations
- Support for the GDB [14] serial remote protocol over TCP/IP. That allows debugging a binary running into the simulator at assembly and/or source level using the GNU debugger (aka GDB)
- Support for a telnet console over the XPS UART Lite

Table 1 shows the simulator memory mapping.
The 1.0 release of the simulator is composed of:

- the simulator source code: unisim-virtex5fxt-1.0.tar.gz
- the present documentation

Please follow the installation instructions in Section 2 to get the simulator building on your own building environment. Section 4 presents the basics for using the simulator. Section 5 presents some examples of use of the simulator. Appendix A contains the technical references (generated) of the simulator.

2 Building the simulator

2.1 Requirements

The following tools or libraries must be installed:

- GNU C++ compiler (aka. g++)
- GNU Flex (aka. flex)
- GNU Bison (aka. bison)
- Standard GNU C++ library (aka. libstdc++-dev that comes with g++)
- Boost (aka. libboost-dev)
- Editline/Libedit (aka. libedit-dev)
- zlib (aka. zlib1g-dev)
- libxml2 (aka. libxml2-dev)

2.2 Installing SystemC 2.2.0 and SystemC TLM 2.0.1

2.2.1 Download the source code

Register at `http://www.systemc.org` and then download systemc-2.2.0.tgz and TLM-2.0.1.tgz from the OSCI standards download page.

2.2.2 Uncompressing the source code tarballs

```
$ tar zxvf systemc-2.2.0.tgz
$ tar zxvf TLM-2.0.1.tgz
```

This will uncompress the source of SystemC in directory systemc-2.2.0 and the source TLM 2.0.1 in directory TLM-2009-07-15

2.2.3 Patching the source code (recommended for g++ $\geq$ 4.1)

Apply the following patch to SystemC 2.2.0: patch-systemc-2.2.0.gz

```
$ cd systemc-2.2.0
$ gzip -dc patch-systemc-2.2.0.gz | patch -p1
```

2.2.4 Configuring SystemC

To configure, compile, and install SystemC in your home directory, do the following at the command prompt:

```
$ cd systemc-2.2.0
$ mkdir objdir
$ cd objdir
$ mkdir ${HOME}/systemc
$ ../configure --prefix=${HOME}/systemc
```

Note: if configure needs automake or autoconf, please install these tools, then rebuild SystemC configure script by running these commands before actually running the configure script:

```
$ cd systemc-2.2.0
$ aclocal
$ automake
```

2.2.5 Compiling and installing SystemC

To handle threads, SystemC relies on QuickThreads, a fast implementation of user's threads. QuickThreads speeds-up threads switching compared to the slower kernel POSIX threads and thus considerably improves overall simulation performance. To compile SystemC with the built-in QuickThreads (recommended), do the following at the command prompt:

```
$ make
$ make install
```

However, if you intend to instrument your simulator (e.g. with valgrind) to debug the simulator memory leaks, bad memory accesses, pointers, and uninitialized memory reads, you should use the slower kernel POSIX threads. To compile SystemC with the kernel POSIX threads, do the following at the command prompt:

```
$ make pthreads
$ make install
```

2.3 Building the UNISIM Virtex 5 FXT simulator

2.3.1 Uncompressing the source code tarball

```
$ tar zxvf unisim-virtex5fxt-1.0.tar.gz
```

2.3.2 Configuring the simulator building process

```
$ cd unisim-virtex5fxt-1.0
$ ./configure \
        --with-systemc=${HOME}/systemc \
        --with-tlm20=${HOME}/TLM-2009-07-15 CXXFLAGS='-O3 -g3 -Wall'
```

2.3.3 Compiling the simulator

```
$ make
```

The simulator binaries are in `virtex5fxt/bin` subdirectory. The simulators comes in four flavors in:

- `unisim-virtex5fxt-1.0`: release simulator without FPU
- `unisim-virtex5fxt-wfpu-1.0`: release simulator with FPU
- `unisim-virtex5fxt-debug-1.0`: developement simulator
- `unisim-virtex5fxt-wfpu-debug-1.0`: developement simulator with FPU

3 Cross-compiling the simulator

This section explains how to build (i.e. cross-compile) the simulator for a host system type (e.g. Windows) from another build system type (e.g. Linux/i386). The simulator is built on the build machine whereas it will run on the host machine. In later sub-sections, we consider cross-compiling the simulator for Windows from a Linux distribution using the mingw32 GCC cross-compiler. Most Linux distributions provide a mingw32 tool chain as a set of packages. Once installed the mingw32 tool chain binary file names are prefixed with:

- `i586-mingw32msvc-` on Ubuntu and Debian Linux distributions
- `i686-pc-mingw32-` on RedHat, Fedora, and SUSE Linux distributions
- `i586-pc-mingw32-` on Mandriva and Mageia Linux distributions

The later sub-sections will refer to the mingw32 tool chain of Ubuntu and Debian Linux distributions.

3.1 Requirements

The following tools must be installed on the Linux build system:

- GNU C++ cross-compiler for the host system type (aka. i586-mingw32msvc-g++)
- GNU Flex (aka. flex)
- GNU Bison (aka. bison)
- Standard GNU C++ library for the host machine (aka. libstdc++)

3.2 Installing a cross-compiled SystemC 2.2.0 and SystemC TLM 2.0.1

3.2.1 Download the source code

Register at `http://www.systemc.org` and then download systemc-2.2.0.tgz and TLM-2.0.1.tgz from the OSCI standards download page.

3.2.2 Uncompressing the source code tarballs

```
$ tar zxvf systemc-2.2.0.tgz
$ tar zxvf TLM-2.0.1.tgz
```

This will uncompress the source of SystemC in directory systemc-2.2.0 and the source TLM 2.0.1 in directory TLM-2009-07-15

3.2.3 Patching the source code (recommended for g++ $\geq$ 4.1)

Apply the following patch to SystemC 2.2.0: patch-systemc-2.2.0.gz

```
$ cd systemc-2.2.0
$ gzip -dc patch-systemc-2.2.0.gz | patch -p1
```

3.2.4 Configuring SystemC

To configure, cross-compile, and install SystemC in your home directory, do the following at the command prompt:

```
$ cd systemc-2.2.0
$ mkdir objdir
$ cd objdir
$ mkdir ${HOME}/systemc-mingw32
$ ../configure --prefix=${HOME}/systemc-mingw32 --host=i586-mingw32msvc
```

Note: if configure needs automake or autoconf, please install these tools, then rebuild SystemC configure script by running these commands before actually running the configure script:

```
$ cd systemc-2.2.0
$ aclocal
$ automake
```

3.2.5 Cross-compiling and installing SystemC

To cross-compile SystemC, do the following at the command prompt:

```
$ make
$ make install
```

3.3 Cross-compiling zlib

Download the source code tarball at `http://zlib.net/zlib-1.2.5.tar.gz`. Uncompress the source code tarball and cross-compile the library:

```
$ tar zxvf zlib-1.2.5.tar.gz
$ cd zlib-1.2.5
$ mkdir ${HOME}/zlib-mingw32
$ make -f win32/Makefile.gcc \
  PREFIX=i586-mingw32msvc- \
  BINARY_PATH=${HOME}/zlib-mingw32/bin \
  INCLUDE_PATH=${HOME}/zlib-mingw32/include \
  LIBRARY_PATH=${HOME}/zlib-mingw32/lib \
  SHARED_MODE=1 install
$ mv ${HOME}/zlib-mingw32/bin/zdll.a ${HOME}/zlib-mingw32/bin/z.dll.a
```

3.4 Cross-compiling libxml2

Download the source code tarball at `ftp://xmlsoft.org/libxml2/libxml2-2.7.8.tar.gz`. Uncompress the source code tarball and cross-compile the library:

```
$ tar zxvf libxml2-2.7.8.tar.gz
$ cd libxml2-2.7.8
$ mkdir ${HOME}/libxml2-mingw32
$ ./configure  --host=i586-mingw32msvc \
      --without-python --with-zlib=${HOME}/zlib-mingw32 \
       CPPFLAGS='-DLIBXML_STATIC'
$ make
$ make install prefix=${HOME}/libxml2-mingw32
```

3.5 Cross-compiling Boost

Download the source code tarball at `http://downloads.sourceforge.net/boost/boost_1_47_0.tar.bz2`. Uncompress the source code tarball and cross-compile the library:

```
$ tar jxvf boost_1_47_0.tar.bz2
$ cd boost_1_47_0
$ mkdir ${HOME}/boost-mingw32
$ ./bootstrap.sh --without-icu
$ MINGW32_VERSION=$(i586-mingw32msvc-g++ -v 2>&1 | tail -1 | awk '{print $3}' | cut -f 1 -d '-')
$ echo "using gcc : ${MINGW32_VERSION} :
              i586-mingw32msvc-g++ :
              <rc>i586-mingw32msvc-windres
              <archiver>i586-mingw32msvc-ar
         ;" > user-config.jam
$ ./bjam toolset=gcc target-os=windows variant=release threading=multi threadapi=win32 \
       link=shared runtime-link=shared --prefix=${HOME}/boost-mingw32 --user-config=user-config.jam \
       --without-mpi --without-python -sNO_BZIP2=1 -sZLIB_BINARY=z.dll \
       -sZLIB_INCLUDE=${HOME}/zlib-mingw32/include -sZLIB_LIBPATH=${HOME}/zlib-mingw32/lib \
       --layout=tagged install
```

3.6 Cross-compiling the UNISIM Virtex 5 FXT simulator

3.6.1 Uncompressing the source code tarball

```
$ tar zxvf unisim-virtex5fxt-1.0.tar.gz
```

3.6.2 Configuring the simulator building process

```
$ cd unisim-virtex5fxt-1.0
$ ./configure.cross \
       --host=i586-mingw32msvc \
       --with-systemc=${HOME}/systemc-mingw32 \
       --with-tlm20=${HOME}/TLM-2009-07-15 \
       --with-zlib=${HOME}/zlib-mingw32 \
       --with-libxml2=${HOME}/libxml2-mingw32 \
       --with-boost=${HOME}/boost-mingw32 \
       CXXFLAGS='-O3 -g3 -Wall'
```

3.6.3 Cross-compiling the simulator

```
$ make -f Makefile.cross
```

The simulator binaries are in `virtex5fxt/bin` subdirectory. The simulators comes in four flavors in:

- `unisim-virtex5fxt-1.0.exe`: release simulator without FPU
- `unisim-virtex5fxt-wfpu-1.0.exe`: release simulator with FPU
- `unisim-virtex5fxt-debug-1.0.exe`: developement simulator
- `unisim-virtex5fxt-wfpu-debug-1.0.exe`: developement simulator with FPU

The simulator binaries may need some DLLs from mingw32 (e.g. `libgcc_s*.dll`) or third party libraries (e.g. `libxml2-2.dll`). Place these DLLs in the `virtex5fxt/bin` subdirectory. If you prefer not to use DLLs, add `LDFLAGS=-static` to the `configure.cross` command line arguments. The simulator binaries can run natively run on a Windows host system, or an emulated Windows using `wine` windows emulator.

4 Getting started

In this section, we present the basics for using the simulator. More details are available in Appendix A.

4.1 Run-time configuration

The simulator has a parametrization system that allows configuring individual simulator components, that is the hardware components, and the services. The simulator stores its configuration (a set of parameters) in a XML configuration file.

The simulator can provide the user with a default XML configuration file with option `-g`:

```
$ unisim-virtex5fxt-wfpu-1.0 -g default_sim_config.xml
```

The XML configuration file can be edited, and then reloaded by the simulator with option `-c`:

```
$ unisim-virtex5fxt-wfpu-1.0 -c sim_config.xml
```

The user can also simply set the value of an individual parameter with option `-s`:

```
$ unisim-virtex5fxt-wfpu-1.0 -s enable-inline-debugger=true
```

The simulator can prints the list of parameter set on the console with option `-l`:

```
$ unisim-virtex5fxt-wfpu-1.0 -l
```

In general, each simulator components have log messages that can be switched on or off using a parameter named `verbose` (or approching):

```
unisim-virtex5fxt-wfpu-1.0 -s cpu.verbose-exception=true
```

The simulator accepts any combination of the above options. For example, you can combine these options to change the value of one or more parameters in an XML configuration file:

```
unisim-virtex5fxt-wfpu-1.0 -c sim_config.xml -s enable-inline-debugger=true -g sim_config.xml
```

4.2 Loading binaries

The simulators has a multi-format loader service that can detect the format of binaries and accordingly instantiate the right loader. The user can set the list of binary files to load in Parameter loader.filename, each filenames being separated by a comma:

```
$ unisim-virtex5fxt-wfpu-1.0 -s loader.filename='boot.elf,vmlinux,device_tree.dtb,initrd.img'
```

In the hypothetic case where the multi-format loader would wrongly guess the format of a binary, the user tells the loader what is the format of the binary file:

```
$ unisim-virtex5fxt-wfpu-1.0 \
   -s loader.filename='boot.elf:elf32,vmlinux:elf32,device_tree.dtb:raw,initrd.img:raw'
```

If, for any reason (virtual memory, self relocation), the simulator must load a binary file to an address that is not the final address indicated in the binary file itself, the user tells the loader to override (when possible) the base address:

```
$ unisim-virtex5fxt-wfpu-1.0 \
     -s loader.filename='boot.elf,vmlinux,device_tree.dtb,initrd.img' \
     -s loader.file1.base-addr=0 \
     -s loader.file1.force-base-addr=true
```

4.3 Serial console

The simulator comes with a UART Lite module on MPLB that the target application can use as a serial console. The simulator telnet service, that is actually a server for the telnet protocol, manages communication between the real network and the virtual UART Lite module. The combination of the UART Lite module, the telnet service and a serial console aware target application, enables using a real telnet client (running on the host machine or any machine on the internet) as virtual serial console.

To enable the serial console, do the following at the command prompt:

```
$ unisim-virtex5fxt-wfpu-1.0 \
    -s enable-telnet=true \
    -s telnet.telnet-tcp-port=1234
```

During initialization, the simulator waits for a telnet client connection on the telnet port. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

At this point, the user can interact with the target application using the telnet client.

4.4 Using the builtin debugger

The simulator has an integrated debugger for debugging the target application an a non-intrusive manner.

To enable the inline-debugger, do the following at the command prompt:

```
$ unisim-virtex5fxt-wfpu-1.0 -s enable-inline-debugger=true
```

The user can enter classical debug commands from the debugger, such as putting breakpoints, watchpoints, stepping instructions, disassembling, dumping memory regions, etc.

To obtain help within the debugger, uses the debugger command `help`:

```
inline-debugger> help
```

4.5 Using a GNU crosstool chain

4.5.1 Building a GNU crosstool chain

A crosstool chain is a tool chain to create and manipulate binary programs for a target architecture and operating system (e.g. PowerPC/Linux) that are different from host architecture and operating system (e.g. x86/Linux) that runs the tool chain. Building a GNU crosstool chain from scratch is a very tedious task. The process consists of a quite high number of undocumented steps, and mostly relies on the users experience. Fortunately a community sharing its experience, the crossgcc mailing list, actively supports a tool, crosstool-NG, that considerably simplifies the process of building a crosstool chain. Xilinx Virtex-5 FXT embedded processor is a PPC440x5 and Xilinx Platform Studio provides a PowerPC hardware FPU. Hence, from now, we will focus our effort on creating a crosstool chain with GNU GCC 4.4.6 (`gcc` and `g++`), GNU C library 2.9 (`glibc`), GNU binutils 2.19.1 (`ld`, `objdump`, and `readelf`), and GNU GDB 6.8 (`gdb`) for target `powerpc-440fp-linux-gnu`.

1. Get crosstool-NG: Download the crosstool-NG archive (e.g crosstool-ng-1.11.3.tar.bz2) from this page: `http://crosstool-ng.org`

2. Uncompress crosstool-NG archive:

   ```
   $ tar jxvf crosstool-ng-1.11.3.tar.bz2
   ```

3. Configure crosstool-NG build:

```
$ cd crosstool-ng-1.11.3
$ ./configure --local
```

4. Compile crosstool-NG

```
$ make
```

5. Configure the crosstool chain build as shown on Figure 2:

```
$ ./ct-ng menuconfig
```

6. Build the crosstool chain (this may take tens of minutes):

```
$ ./ct-ng build
```

7. You crosstool chain is ready in `${HOME}/crosstool/powerpc-440fp-linux-gnu`

```
Paths and misc options  --->
(${HOME}/crosstool/powerpc-440fp-linux-gnu) Prefix directory

Target options  --->
        Target Architecture (powerpc)  --->
    (440fp) Emit assembly for CPU
    (440fp) Tune for CPU
        Floating point: (hardware (FPU))  --->

Toolchain options  --->
    (440fp) Tuple's vendor string

Operating System  --->
    Target OS (linux)  --->
    Linux kernel version (2.6.35.13 (longterm))  --->

Binary utilities  --->
    binutils version (2.19.1)  --->

    [*] binutils libraries for the target
    [*]   libiberty (NEW)
    [*]   libbfd (NEW)

C compiler  --->
    gcc version (4.4.6)  --->
    [*] C++
    [ ] Link libstdc++ statically into the gcc binary

C-library  --->
    C library (glibc)  --->
    glibc version (2.9)  --->
    (-U_FORTIFY_SOURCE) extra target CFLAGS
    [*] Use the ports addon

Debug facilities  --->
    [*] gdb  --->

Companion libraries  --->
    MPFR version (2.4.2)  --->
    CLooG/ppl version (0.15.9)  --->
```

Figure 2: crosstool chain build configuration (changes from default only) for Crosstool-NG 1.11.3.

4.5.2 Using cross-GCC

A GCC cross-compiler together with a LD cross-linker can be used to create binaries for the target machine from the host machine:

```
$ ${HOME}/crosstool/powerpc-440fp-linux-gnu/bin/powerpc-440fp-linux-gnu-gcc \
                  nodefaultlibs -nostdlib -mcpu=440fp -c hello.c -o hello.o
$ ${HOME}/crosstool/powerpc-440fp-linux-gnu/bin/powerpc-440fp-linux-gnu-ld hello.lds -o hello.elf
```

Simulator can directly loads such binary files, see Section 4.2.

4.5.3 Using cross-GDB

GNU GDB client can debug applications running on a remote/local host on the network. The application is run under the control of program `gdbserver` while program `gdb` only manages interactions with the user. Program `gdbserver` and `gdb` communicates over the TCP/IP network using a documented serial remote protocol. The simulator has a GDB server service that implements the GDB serial remote protocol, so that the simulator can acts as Program `gdbserver` from the GDB client point of view.

To enable the simulator GDB server, do the following at the command prompt:

```
$ unisim-virtex5fxt-wfpu-1.0 -s enable-gdb-server=true -s gdb-server.tcp-port=1234
```

During initialization, the simulator waits for a GDB client connection on the GDB server TCP/IP port. In another console, connect a GDB client to the simulator:

```
$ powerpc-440fp-linux-gnu-gdb boot.elf
(gdb) target remote :1234
```

5 Examples of use

In this section, we present some examples of use for the simulator. We provide you with scripts and makefiles to build in an automatic manner all the examples. These examples are available for download in source and binary forms on our web site (`http://unisim-vp.org`):

- Basic system level unit tests: [hyperlink to binaries], [hyperlink to sources]
- MiBench version 1: [hyperlink to binaries], [hyperlink to sources]
- light weight Linux distribution: [hyperlink to binaries], [hyperlink to sources]

5.1 Building the examples

To build the examples you need a functional cross tool-chain for target for target `powerpc-440fp-linux-gnu`, see Section 4.5.1. In later sub-sections, we assume that your cross tool-chain is installed in `/opt/crosstool/gcc-4.4.6-glibc-2.9/powerpc-440fp-linux-gnu`

5.1.1 Building the basic system level unit tests

From the directory where you uncompressed the archive, do the following at the command prompt:

```
$ make CROSS_COMPILE=/opt/crosstool/gcc-4.4.6-glibc-2.9/powerpc-440fp-linux-gnu/bin/powerpc-440fp-linux-gnu-
```

5.1.2 Building the MiBench version 1 benchmarks

From the directory where you uncompressed the archive, do the following at the command prompt:

```
./build.sh all /opt/crosstool/gcc-4.4.6-glibc-2.9/powerpc-440fp-linux-gnu
```

5.1.3 Building the light weight Linux distribution

From the directory where you uncompressed the archive, do the following at the command prompt:

```
./build.sh all /opt/crosstool/gcc-4.4.6-glibc-2.9/powerpc-440fp-linux-gnu
```

5.2 Basic system level tests

Table 2 shows a summary of basic system tests.

Test Name	Directory	IntC	Timer	UART Lite	NOR Flash
Echo	`xps_uart_lite/echo`	√		√	
Poll	`xps_timer/poll`		√	√	
Gen	`xps_timer/gen`	√	√	√	
Cap	`xps_timer/cap`	√	√	√	
PWM	`xps_timer/pwm`	√	√	√	
Autoselect	`s29gl256p/autoselect`	√		√	√
CFI query	`s29gl256p/cfi_query`	√		√	√
Chip erase	`s29gl256p/chip_erase`	√		√	√
Sector erase	`s29gl256p/sector_erase`	√		√	√
Single word programming	`s29gl256p/single_word_programming`	√		√	√
Unlock bypass chip erase	`s29gl256p/unlock_bypass_chip_erase`	√		√	√
Unlock bypass sector erase	`s29gl256p/unlock_bypass_sector_erase`	√		√	√
Unlock bypass word programming	`s29gl256p/unlock_bypass_word_programming`	√		√	√
Write buffer programming	`s29gl256p/write_buffer_programming`	√		√	√

Table 2: Summary of basic system level tests.

5.2.1 Echo

The test reads characters from the serial console. It prints the read characters on the serial console.

1. At the command prompt, do the following:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c xps_uart_lite/echo/sim_config.xml
```

2. In another console, do the following at command prompt:

```
$ telnet localhost 1234
```

3. Press some keys in the telnet console. The corresponding characters should prints in the telnet console as you type

5.2.2 Poll

The test polls the timer/counter #0. It prints some of the sampled values on the serial console. To run the test within the simulator:

1. Enter Directory `basic_system`, start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c xps_timer/poll/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Time is 6 us
Time is 116 us
Time is 227 us
Time is 338 us
Time is 449 us
Time is 560 us
Time is 671 us
Time is 782 us
Time is 893 us
Time is 1004 us
Time is 1115 us
Time is 1226 us
Time is 1337 us
Time is 1448 us
Time is 1559 us
Time is 1670 us
Time is 1782 us
Time is 1893 us
Time is 2004 us
Time is 2115 us
Time is 2226 us
Time is 2337 us
Time is 2448 us
Time is 2559 us
Time is 2670 us
```

```
Time is 2782 us
Time is 2894 us
Time is 3005 us
Time is 3116 us
Time is 3228 us
Time is 3340 us
Time is 3451 us
Time is 3562 us
Time is 3673 us
Time is 3785 us
Time is 3897 us
Time is 4008 us
Time is 4119 us
Time is 4230 us
Time is 4342 us
Test is finished
```

5.2.3 Gen

The test uses the timer generate mode with interrupt generation every 100 μs. It prints the tick of timer on the serial console.
To run the test within the simulator:

1. Enter Directory `basic_system`, start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c xps_timer/gen/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Tick: 100 us
Tick: 200 us
Tick: 300 us
Tick: 400 us
Tick: 500 us
Tick: 600 us
Tick: 700 us
Tick: 800 us
Tick: 900 us
Tick: 1000 us
Tick: 1100 us
Tick: 1200 us
Tick: 1300 us
Tick: 1400 us
Tick: 1500 us
Tick: 1600 us
....

....
Tick: 8300 us
Tick: 8400 us
Tick: 8500 us
Tick: 8600 us
Tick: 8700 us
Tick: 8800 us
Tick: 8900 us
Tick: 9000 us
```

```
Tick: 9100 us
Tick: 9200 us
Tick: 9300 us
Tick: 9400 us
Tick: 9500 us
Tick: 9600 us
Tick: 9700 us
Tick: 9800 us
Tick: 9900 us
Tick: 10000 us
Test is finished
```

5.2.4 Cap

The test uses the timer capture mode (randomized input between 1 μs and 3.995 μs). It prints the captured time stamp on the serial console.
To run the test within the simulator:

1. Enter Directory `basic_system`, start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c xps_timerp/cap/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Last capture is 360 ns
Last capture is 23090 ns
Last capture is 45930 ns
Last capture is 69285 ns
Last capture is 93040 ns
Last capture is 117545 ns
Last capture is 143545 ns
Last capture is 169325 ns
Last capture is 195480 ns
Last capture is 220590 ns
Last capture is 244635 ns
Last capture is 270840 ns
Last capture is 294800 ns
Last capture is 319990 ns
Last capture is 348385 ns
Last capture is 370810 ns
Last capture is 397785 ns
Last capture is 422900 ns
Last capture is 449070 ns
Last capture is 473000 ns
Last capture is 497565 ns
Last capture is 521180 ns
Last capture is 547700 ns
Last capture is 572140 ns
Last capture is 596700 ns
Last capture is 622295 ns
Last capture is 647230 ns
Last capture is 671745 ns
Last capture is 697080 ns
Last capture is 722645 ns
Last capture is 746020 ns
Last capture is 771475 ns
```

```
Last capture is 797765 ns
Last capture is 818050 ns
Last capture is 847275 ns
Last capture is 872910 ns
Last capture is 897765 ns
Last capture is 922820 ns
Last capture is 947330 ns
Last capture is 972370 ns
Last capture is 998170 ns
Last capture is 1024420 ns
Test is finished
```

5.2.5 PWM

The test uses the timer in PWM (Pulse Width Modulation) mode with a period of 2 μs and a duty cycle of 300 ns. To run the test within the simulator:

1. Enter Directory `basic_system`, start the simulator and filter PWM0 activity:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c xps_timer/pwm/sim_config.xml \
                             -s timer.verbose=true | grep 'PWM0 signal'
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
You should observe a PWM output with a period of 2000 ns, and a duty cycle of 300 ns.

timer: 7815 ns: PWM0 signal goes high
timer: 8115 ns: PWM0 signal goes low
timer: 9815 ns: PWM0 signal goes high
timer: 10115 ns: PWM0 signal goes low
timer: 11815 ns: PWM0 signal goes high
timer: 12115 ns: PWM0 signal goes low
timer: 13815 ns: PWM0 signal goes high
timer: 14115 ns: PWM0 signal goes low
timer: 15815 ns: PWM0 signal goes high
timer: 16115 ns: PWM0 signal goes low
timer: 17815 ns: PWM0 signal goes high
timer: 18115 ns: PWM0 signal goes low
timer: 19815 ns: PWM0 signal goes high
timer: 20115 ns: PWM0 signal goes low
timer: 21815 ns: PWM0 signal goes high
timer: 22115 ns: PWM0 signal goes low
timer: 23815 ns: PWM0 signal goes high
timer: 24115 ns: PWM0 signal goes low
timer: 25815 ns: PWM0 signal goes high
timer: 26115 ns: PWM0 signal goes low
timer: 27815 ns: PWM0 signal goes high
timer: 28115 ns: PWM0 signal goes low
timer: 29815 ns: PWM0 signal goes high
timer: 30115 ns: PWM0 signal goes low
timer: 31815 ns: PWM0 signal goes high
timer: 32115 ns: PWM0 signal goes low
timer: 33815 ns: PWM0 signal goes high
timer: 34115 ns: PWM0 signal goes low
....
```

5.2.6 Autoselect

The test puts the S29GL256P NOR Flash chip in autoselect mode. It reads the manufacturer and device IDs. It prints the two IDs on the serial console. It prints also protection status of each sector.
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/autoselect/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Write unlock cycle 1
Write unlock cycle 2
Write autoselect command
Manufacturer ID: 0x0001
Device ID word 1: 0x227e
Device ID word 2: 0x2222
Device ID word 3: 0x2201
Protection status (U=unprotected/P=protected): UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
Exit autoselect
Test is finished
```

5.2.7 CFI query

The test puts the S29GL256P NOR Flash chip in CFI (Common Flash Interface) query mode. It queries the unique ASCII string "QRY". For each of the three characters, it prints on the serial console whether they match what is expected ("QRY").
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/cfi_query/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Write CFI entry command
Read Q: OK
Read R: OK
Read Y: OK
Exit CFI query mode (Write reset command)
Test is finished
```

5.2.8 Chip erase

The test erases the S29GL256P NOR Flash chip. It checks that all sectors have been erased.
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/chip_erase/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Write unlock cycle 1
Write unlock cycle 2
Write setup command
Write additional unlock cycle 1
Write additional unlock cycle 2
Write chip erase command
Verifying sectors..............................................................
..............................................................................
..............................................................................
................................
Done
Test is finished
```

5.2.9 Sector erase

The test erases the S29GL256P NOR Flash chip, one sector at a time. It checks that all sectors have been erased.
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/sector_erase/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Erasing sector #0 of 256
Write unlock cycle 1
Write unlock cycle 2
Write setup command
Write additional unlock cycle 1
Write additional unlock cycle 2
Write sector erase command
Verifying sector: OK
....

....
Erasing sector #255 of 256
Write unlock cycle 1
Write unlock cycle 2
Write setup command
Write additional unlock cycle 1
Write additional unlock cycle 2
Write sector erase command
Verifying sector: OK
```

5.2.10 Single word programming

The test program a word on the S29GL256P NOR Flash chip. It verifies that word has been effectively programmed.
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/single_word_programming/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Write unlock cycle 1
Write unlock cycle 2
Write program setup command
Write data to be programmed
Verify programmed data: OK
```

5.2.11 Unlock bypass chip erase

The test erases the S29GL256P NOR Flash chip four times. It checks that all sectors have been erased.
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/unlock_bypass_chip_erase/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Write unlock cycle 1
Write unlock cycle 2
Write unlock bypass command
=== Pass #0 ===
Write setup command
Write chip erase command
Verifying sectors.....................................................................
.....................................................................
.....................................................................
.....................
Done
=== Pass #1 ===
Write setup command
Write chip erase command
Verifying sectors.....................................................................
.....................................................................
.....................................................................
.....................
```

```
Done
=== Pass #2 ===
Write setup command
Write chip erase command
Verifying sectors.....................................................
.....................................................................
.....................................................................
.............................
Done
=== Pass #3 ===
Write setup command
Write chip erase command
Verifying sectors.....................................................
.....................................................................
.....................................................................
.............................
Done
Exit unlock bypass
Test is finished
```

5.2.12 Unlock bypass sector erase

The test erases the S29GL256P NOR Flash chip, one sector at a time. It checks that all sectors have been erased.
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/unlock_bypass_sector_erase/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Write unlock cycle 1
Write unlock cycle 2
Write unlock bypass command
Erasing sector #0 of 256
Write setup command
Write sector erase command for sector #0
Verifying sector: OK
....

....
Erasing sector #255 of 256
Write setup command
Write sector erase command for sector #255
Verifying sector: OK
Exit unlock bypass
Test is finished
```

5.2.13 Unlock bypass word programming

The test programs four words in the S29GL256P NOR Flash chip, one word at a time. It verifies that the four words have beed effectively programmed.
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/unlock_bypass_sector_erase/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Write unlock cycle 1
Write unlock cycle 2
Write unlock bypass command
Write program setup command
Write data #0 to be programmed (0x4321)
Write program setup command
Write data #1 to be programmed (0x8765)
Write program setup command
Write data #2 to be programmed (0x9087)
Write program setup command
Write data #3 to be programmed (0x3852)
Re-reading and Verifying programmed data: OK (0x4321)
Re-reading and Verifying programmed data: OK (0x8765)
Re-reading and Verifying programmed data: OK (0x9087)
Re-reading and Verifying programmed data: OK (0x3852)
Exit unlock bypass
Test is finished
```

5.2.14 Write buffer programming

The test programs four words in the S29GL256P NOR Flash chip, using the write buffer. It verifies that the four words have beed effectively programmed.
To run the test within the simulator:

1. Enter Directory `basic_system`, and start the simulator:

```
$ cd basic_system
$ unisim-virtex5fxt-wfpu-1.0 -c s29gl256p/unlock_bypass_sector_erase/sim_config.xml
```

2. In another console, connect a telnet client to the simulator:

```
$ telnet localhost 1234
```

3. The output of the test on the telnet console should be:

```
Write unlock cycle 1
Write unlock cycle 2
Write write buffer load command
Write write word count (minus 1)
Write data word #0 (0x4321)
Write data word #1 (0x8765)
Write data word #2 (0x9087)
Write data word #3 (0x3852)
Write confirm command
Re-reading and verifying word #0: OK (0x4321)
Re-reading and verifying word #1: OK (0x8765)
Re-reading and verifying word #2: OK (0x9087)
Re-reading and verifying word #3: OK (0x3852)
Test is finished
```

5.3 A light weight Linux distribution

The simulator can easily run a minimalist Linux distribution for a Xilinx Virtex-5 FXT development board. Table 3 shows the files to load in the simulator to run such a minimalist Linux distribution. In later sub-sections, we explain how these files are obtained. We provides you with prebuilt files to allow you quickly boot a minimalist Linux distribution within the simulator. Go directly to Sub-section 5.3.7 if you don't bother about the technique behind creating a light weight Linux distribution for the simulator. To build such a minimalist Linux distribution, you need a working crosstool chain, see Section 4.5.1.

File	Start address	Memory	Description
`boot.elf`	`0xffff0000`	BRAM	Boot program
`vmlinux`	`0x00000000`	RAM	Linux kernel
`device_tree.dtb`	`0x00800000`	RAM	Device Tree
`initrd.img`	`0x00900000`	RAM	Initial RAM disk

Table 3: Files for booting Linux in the simulator.

5.3.1 The boot program

The boot program is loaded in BRAM which is behind the MPLB. At reset, the processor starts executing instructions from physical address `0xfffffffc`. Thus the boot program is located in memory so that it has an instruction (usually a branch) at that physical address. The role of the boot program is to initialize early boot and provides the Kernel with some parameters in the processor registers. Table 4 shows the registers that needs to be initialized before branching into the Linux Kernel entry point.

The boot program, which source is shown on Figure 3, has been designed to make instruction at Label `start` match the processor start address. Actually, a branch instructions to elsewhere in the boot program (Label `init`) is placed at the reset address. The boot program then starts enabling the MCI (Memory Controller Interface) so that processor can use RAM which is behind the MCI. It programs the MMU to map the whole 256 MB RAM in the processor address space. It initializes required register parameters of the Linux kernel, that is the device tree address into Register `r3`, the initial RAM disk address range into Registers `r4` and `r5`, the Linux kernel start address into Register `SRR0`, and the value of MSR into Register `SRR1`. It then branches to the Linux kernel using an `rfi` instruction (return from interrupt).

5.3.2 The Linux kernel

The Linux kernel (aka. `vmlinux`) must be configured and built for the target platform. The steps to follow are:

1. Get the Linux kernel source code: Download the `linux-2.6.38.7.tar.bz2` tarball at `ftp://ftp.kernel.org/pub/linux/kernel/v2.6/linux-2.6.38.7.tar.bz2`.

2. Uncompress the tarball

```
$ tar jxvf linux-2.6.38.7.tar.bz2
```

3. Before starting configuring the Linux kernel build, we need to modify a Linux configuration file to enable support of hardware FPU. In File `arch/powerpc/platforms/44x/Kconfig`, after:

Registers	Description
r3	Address of the device tree
r4	Start address of the initial RAM disk
r5	End address of the initial RAM disk

Table 4: Linux kernel register parameters.

```
 1 #include "ppc44x_regs.h"
 2 #define KERNEL_START 0x00000000
 3 #define DEV_TREE_PTR 0x00800000
 4 #define INITRD_START 0x00900000
 5 #define INITRD_END 0x01900000
 6 #define TLB0 0x00000250  /* map 0x00000000 (256 MB) in system address space */
 7 #define TLB1 0x0         /* at physical address 0x0 */
 8 #define TLB2 0x7         /* with access control SX=1 SW=1 SR=1 */
 9
10 #define MI_CONTROL 0x10
11 #define MI_CONTROL_ENABLE 0x8000
12
13   .section ".text"
14   .org 0
15
16 init:
17   mfdcr r0, MI_CONTROL
18   oris r0, r0, MI_CONTROL_ENABLE
19   mtdcr MI_CONTROL, r0    /* enable MCI */
20   li r0, 0          /* TLB entry #0 */
21   lis r8, TLB0@h
22   ori r8, r8, TLB0@l
23   li r9, 0
24   mtspr MMUCR, r9   /* MMUCR[STID] = 0 */
25   lis r10, TLB1@h
26   ori r10, r10, TLB1@l
27   lis r11, TLB2@h
28   ori r11, r11, TLB2@l
29   tlbwe r8, r0, 0   /* Set EPN, V, TS, SIZE, TID of UTLB entry #0 word #0 */
30   tlbwe r10, r0, 1  /* Set RPN, ERPN of UTLB entry #0 word #1 */
31   tlbwe r11, r0, 2  /* Set U0 U1 U2 U3 W I M G E UX UW UR SX SW SR of UTLB entry #0 word #2 */
32   lis r3, DEV_TREE_PTR@h
33   ori r3, r3, DEV_TREE_PTR@l /* r3 <- device tree address */
34   lis r4, INITRD_START@h
35   ori r4, r4, INITRD_START@l /* r4 <- initrd start */
36   lis r5, INITRD_END@h
37   ori r5, r5, INITRD_END@l  /* r5 <- initrd end */
38   li r6, 0 /* r6 <- start of kernel command line (unused as kernel command line is in dev tree) */
39   li r7, 0 /* r7 <- end of kernel command line (unused as kernel command line is in dev tree) */
40   lis r0, KERNEL_START@h
41   ori r0, r0, KERNEL_START@l
42   mtspr SRR0, r0    /* SRR0 <- kernel start address */
43   li r1, 0
44   mtspr SRR1, r1    /* MSR <- 0 */
45   rfi   /* Branch to Linux kernel and invalidate shadow TLBs */
46
47   .global  start
48   .org 0xffc
49 start:
50   b init
```

Figure 3: Boot program (boot.S compiled as boot.elf) loaded in BRAM.

```
config XILINX_VIRTEX_5_FXT
       bool
       select XILINX_VIRTEX
```

add the following line:

```
       select PPC_FPU
```

4. We can now configure the Linux kernel build:

```
$ make \
    ARCH=powerpc \
    CROSS_COMPILE=${HOME}/crosstool/powerpc-440fp-linux-gnu/bin/powerpc-440fp-linux-gnu- \
    V=1 menuconfig
```

   The Linux kernel should be configured as shown on Figure 4. Once configuration is finished, at exit don't forget to answer "Yes" when prompted for saving the settings in File `.config`.

5. The Linux kernel (`vmlinux`) can now be built. The build may take tens of minutes. Do the following at the command prompt:

```
$ make \
    ARCH=powerpc \
    CROSS_COMPILE=${HOME}/crosstool/powerpc-440fp-linux-gnu/bin/powerpc-440fp-linux-gnu- \
    V=1 vmlinux
```

5.3.3 The device Tree

On PowerPC embedded platforms, the Linux kernel uses a hierarchical list of devices, namely a device tree where leaves are devices and non-leaf nodes are buses, bridges and interconnects. It is a rather detailed machine description to allow the Linux kernel to correctly initialize devices and route interrupts to the interrupt routines. The boot loader or anything else launching the Linux kernel should provide the Linux kernel with the device tree. Linux provides a device tree compiler that compile a text description (`.dts` file) in a loadable binary form (`.dtb` file). Our example provides user with three device trees, that is one for each initial RAM disk size (16 MB, 32 MB and 64 MB). The provided device trees are:

1. `device-tree-16m.dtb`: device tree for a 16 MB large initial RAM disk
2. `device-tree-32m.dtb`: device tree for a 32 MB large initial RAM disk
3. `device-tree-64m.dtb`: device tree for a 64 MB large initial RAM disk

5.3.4 The initial RAM disk

The initial RAM disk file is a file containing an image of the root file system. We consider creating an initial RAM disk with an Ext2 file system in it. Note that creating an image requires root privileges because mounting a file system requires root privileges.

The steps to follow are:

1. Create an empty image (`initrd.img`) of 16 MB:

```
[root@localhost] $ dd if=/dev/zero of=initrd.img count=16384 bs=1024
```

2. Create an `Ext2` file system in image:

```
Processor support  --->
        Processor Types (AMCC 44x, 46x or 47x)

General Setup  --->
    [*] Initial RAM filesystem and RAM disk (initramfs/initrd) support

Platform support  --->
    [*] Generic Xilinx Virtex 5 FXT board support

Device Drivers  --->
    [*] Block devices  --->
        (16)    Default number of RAM disks
         (65536) Default RAM disk size (kbytes)

        Character devices  --->
            Serial drivers  --->
                <*> Xilinx uartlite serial port support
                [*]   Support for console on Xilinx uartlite serial port

    [*] GPIO Support  --->
        [*] Xilinx GPIO support

    [*] Watchdog Timer Support  --->
        <*> PowerPC Book-E Watchdog Timer

File systems  --->
    <*> Second extended fs support

Kernel hacking  --->
    [*] Compile the kernel with debug info
```

Figure 4: Linux kernel configuration (changes from default only).

```
[root@localhost] $ mke2fs -F -m 0 initrd.img
```

3. Create a mount point and mount image on it:

```
[root@localhost] $ mkdir /media/initrd
[root@localhost] $ mount -o loop initrd.img /mnt/initrd
```

4. Now you can directly access withing the image using directory `/mnt/initrd`. In directory `/mnt/initrd`, copy every files and directories you want in your image.

5. Unmount image:

```
[root@localhost] $ umount /media/initrd
```

We provide several prebuilt initial RAM disks. Some of the initial RAM disks contain a MiBench benchmark (see Table 6) in `/opt`. Table 5 summarizes the prebuilt initial RAM disks.

Initial RAM disk	**Content**
`initrd.img`	Busybox only
`initrd_automotive_basicmath.img`	Busybox + automotive/basicmath
`initrd_consumer_typeset.img`	Busybox + consumer/typeset
`initrd_office_rsynth.img`	Busybox + office/rsynth
`initrd_telecomm_adpcm.img`	Busybox + telecomm/ADPCM
`initrd_automotive_bitcount.img`	Busybox + automotive/bitcount
`initrd_office_stringsearch.img`	Busybox + office/stringsearch
`initrd_telecomm_crc32.img`	Busybox + telecomm/CRC32
`initrd_automotive_qsort.img`	Busybox + automotive/qsort
`initrd_network_dijkstra.img`	Busybox + network/dijkstra
`initrd_security_blowfish.img`	Busybox + security/blowfish
`initrd_telecomm_fft.img`	Busybox + telecomm/FFT
`initrd_automotive_susan.img`	Busybox + automotive/susan
`initrd_network_patricia.img`	Busybox + network/patricia
`initrd_security_pgp.img`	Busybox + security/PGP
`initrd_telecomm_gsm.img`	Busybox + telecomm/GSM
`initrd_consumer_jpeg.img`	Busybox + consumer/JPEG MiBench
`initrd_office_ghostscript.img`	Busybox + office/ghostscript
`initrd_security_rijndael.img`	Busybox + security/rijndael
`initrd_consumer_lame.img`	Busybox + consumer/lame
`initrd_office_ispell.img`	Busybox + office/ispell
`initrd_security_sha.img`	Busybox + security/sha

Table 5: Initial RAM disks.

5.3.5 Busybox

Busybox is a light weight shell that includes most of standard UNIX commands. Actually commands are symbolic links to the busybox binary. Busybox knows which commands to implement looking at its `argv[0]`. It is usually a good idea to install Busybox in the initial RAM disk image of an Linux-based embedded platform because it's tiny and easy to cross-compile. The Linux kernel (`vmlinux`), at the end of the boot procedure, starts `linuxrc` located at the root of the root file system (e.g. in an initial RAM disk on device `/dev/ram0`). Usually `linuxrc` is a symbolic link to the boot shell (e.g. the Busybox ash shell located at `/bin/ash`).

5.3.6 The MiBench version 1 benchmarks

The MiBench [13, 1] version 1 is a free, commercially representative embedded benchmark suite. Most of the benchmarks come with small and large data sets. Table 6 summarizes the available benchmarks and there current status. Be aware that some of the benchmarks are so difficult to cross-compile (they were only intended to be natively compiled on the machine that will run them) that we gave up to build them. Also note that some benchmarks do not run correctly on PowerPC processors because these benchmarks wrongly assumes endian (PowerPC processors natural endian is big-endian whether processors of standard PC are little-endian). When such limitations exist, they are explained in the status column of the table.

5.3.7 Running the example

A very small Linux distribution based on Busybox together with a configuration file are provided for the simulator. To boot that small Linux distribution within the simulator, do the following at the command prompt:

```
$ cd linux_distro
$ unisim-virtex5fxt-1.0-wfpu-1.0 -c sim_config.xml
```

The simulator acts as a telnet server to emulate a terminal over the XPS UART Lite. Once started, it waits for a telnet client connection like `telnet` or `PuTTY`. To connect the telnet client to the simulator, do the following at the command prompt:

```
$ telnet localhost 1234
```

You can see (in the telnet terminal) the Linux distribution booting in the simulator:

```
[    0.000000] Using Xilinx Virtex440 machine description
[    0.000000] Linux version 2.6.38.7 (gmouchard@is010192.saclay.cea.fr) (gcc version
4.4.6 (crosstool-NG-1.11.3) ) #1 PREEMPT Thu Sep 8 10:28:23 CEST 2011
[    0.000000] Found initrd at 0xc0900000:0xc1900000
[    0.000000] Zone PFN ranges:
[    0.000000]   DMA      0x00000000 -> 0x00001000
[    0.000000]   Normal   empty
[    0.000000] Movable zone start PFN for each node
[    0.000000] early_node_map[1] active PFN ranges
[    0.000000]     0: 0x00000000 -> 0x00001000
[    0.000000] MMU: Allocated 1088 bytes of context maps for 255 contexts
[    0.000000] Built 1 zonelists in Zone order, mobility grouping off.  Total pages: 4094
[    0.000000] Kernel command line: root=/dev/ram0 rw init=linuxrc console=ttyUL0
[    0.000000] PID hash table entries: 1024 (order: -4, 4096 bytes)
[    0.000000] Dentry cache hash table entries: 32768 (order: 1, 131072 bytes)
[    0.000000] Inode-cache hash table entries: 16384 (order: 0, 65536 bytes)
[    0.000000] Memory: 240192k/262144k available (3712k kernel code, 21952k reserved,
384k data, 689k bss, 256k init)
[    0.000000] Kernel virtual memory layout:
[    0.000000]   * 0xfffd0000..0xffff0000  : fixmap
[    0.000000]   * 0xfde00000..0xfe000000  : consistent mem
```

Benchmark	Category	Description	Status
sha	security	160-bit secure hash algorithm	OK
blowfish	security	Blowfish encryption and decryption	OK
rijndael	security	AES encryption and decryption	OK after patching aesxam.c (use ftell instead of fgetpos)
pgp	security	Asymetric (public key) encryption and decryption	OK
qsort	automotive	Sorting algorithm	OK
susan	automotive	Corner and edge recognition	OK
basicmath	automotive	Solving cubic polynomial, computing integer square root and converting angles	OK
bitcount	automotive	Count set bits in integer	OK
dijkstra	network	Shortest path in a graph	OK
FFT	telecomm	Fast Fourier Transform	OK
GSM	telecomm	Global System for Mobile Communications encoder and decoder	OK
CRC32	telecomm	32-bit Cyclic Redundancy Check	OK
ADPCM	telecomm	Adaptative Differential Pulse Code Modulation encoder and decoder	OK with the small data set (a RAM disk of 64 MB is not sufficient to run the benchmark with the large data set)
typeset	consumer	A batch document formatter	OK
lame	consumer	MP3 encoder	OK
mad	consumer	MPEG audio decoding	Can't get it to cross-compile
JPEG	consumer	JPEG encoder and decoder	OK
tiff	office	Conversion and Dithering of pictures	Can't get it to cross-compile
ghostscript	office	Postscript renderer	Crashes on PowerPC
ispell	office	Spell checker	OK
stringsearch	office	Search for words in text	OK
rsynth	office	Text to speech synthesis	OK
sphinx	office	Speech decoder	Can't be compiled on modern compilers

Table 6: MiBench version 1.0.

```
[    0.000000]   * 0xfde00000..0xfde00000  : early ioremap
[    0.000000]   * 0xd1000000..0xfde00000  : vmalloc & ioremap
[    0.000000] Preemptable hierarchical RCU implementation.
[    0.000000]  RCU-based detection of stalled CPUs is disabled.
[    0.000000]  Verbose stalled-CPUs detection is disabled.
[    0.000000] NR_IRQS:512
[    0.000000] clocksource: timebase mult[a00000] shift[22] registered
[    0.000000] Console: colour dummy device 80x25
[    0.000343] pid_max: default: 32768 minimum: 301
[    0.000814] Mount-cache hash table entries: 8192
[    0.007584] NET: Registered protocol family 16
[    0.009029] PCI: Probing PCI hardware
[    0.017297] bio: create slab <bio-0> at 0
[    0.017779] XGpio: /plb@0/gpio@81460000: registered
[    0.017978] XGpio: /plb@0/gpio@81400000: registered
[    0.018185] XGpio: /plb@0/gpio@81420000: registered
[    0.018389] XGpio: /plb@0/gpio@81440000: registered
[    0.018872] vgaarb: loaded
[    0.019767] Switching to clocksource timebase
[    0.035481] NET: Registered protocol family 2
[    0.035774] IP route cache hash table entries: 16384 (order: 0, 65536 bytes)
[    0.037037] TCP established hash table entries: 8192 (order: 0, 65536 bytes)
[    0.037372] TCP bind hash table entries: 8192 (order: -1, 32768 bytes)
[    0.037569] TCP: Hash tables configured (established 8192 bind 8192)
[    0.037584] TCP reno registered
[    0.037601] UDP hash table entries: 4096 (order: 0, 65536 bytes)
[    0.037941] UDP-Lite hash table entries: 4096 (order: 0, 65536 bytes)
[    0.038918] NET: Registered protocol family 1
[    0.039363] RPC: Registered udp transport module.
[    0.039379] RPC: Registered tcp transport module.
[    0.039392] RPC: Registered tcp NFSv4.1 backchannel transport module.
[    0.039571] Trying to unpack rootfs image as initramfs...
[    0.039920] rootfs image is not initramfs (junk in compressed archive); looks like an initrd
[    0.086721] Freeing initrd memory: 16384k freed
[    0.091563] ROMFS MTD (C) 2007 Red Hat, Inc.
[    0.091826] msgmni has been set to 500
[    0.092236] io scheduler noop registered
[    0.092248] io scheduler deadline registered
[    0.092332] io scheduler cfq registered (default)
[    0.265669] 84000000.serial: ttyUL0 at MMIO 0x84000000 (irq = 19) is a uartlite
[    0.266450] console [ttyUL0] enabled
[    0.267903] brd: module loaded
[    0.271850] loop: module loaded
[    0.272234] Xilinx SystemACE device driver, major=254
[    0.273794] mousedev: PS/2 mouse device common for all mice
[    0.274285] PowerPC Book-E Watchdog Timer Loaded
[    0.275977] TCP cubic registered
[    0.275999] NET: Registered protocol family 17
[    0.277548] RAMDISK: ext2 filesystem found at block 0
[    0.277586] RAMDISK: Loading 16384KiB [1 disk] into ram disk... done.
[    0.614251] VFS: Mounted root (ext2 filesystem) on device 1:0.
[    0.614601] Freeing unused kernel memory: 256k init
starting pid 741, tty '': '/etc/init.d/rcS'
starting pid 744, tty '': '/bin/cttyhack /etc/init.sh'


BusyBox v1.18.4 (2011-09-08 10:28:36 CEST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

[/]$
```

Once the boot is over, you can enter shell commands at the command prompt in the telnet terminal. For example, you can ask for CPU informations to the Linux kernel by doing the following at the command prompt in the telnet terminal:

```
[/]$ cat /proc/cpuinfo
processor       : 0
cpu             : 440 in Virtex-5 FXT
clock           : 400.000000MHz
revision        : 25.18 (pvr 7ff2 1912)
bogomips        : 800.00
timebase        : 400000000
platform        : Xilinx Virtex440
model           : testing
Memory          : 256 MB
[/]$
```

You may also want to try some MiBench benchmarks within that small Linux distribution. We provide an initial RAM disk for almost each MiBench benchmark and the corresponding simulator configuration file. For example, to run the MiBench Lame benchmark, do the following at the command prompt:

```
$ cd linux_distro
$ unisim-virtex5fxt-1.0-wfpu-1.0 -c sim_config_consumer_lame.xml
```

In another console, connect the telnet client to the simulator:

```
$ telnet localhost 1234
```

Run the MiBench Lame benchmark:

```
[/]$ cd /opt/consumer/lame
[/opt/consumer/lame]$ PATH=$(pwd) ./runme_large.sh
LAME version 3.70 (www.sulaco.org/mp3)
GPSYCHO: GPL psycho-acoustic and noise shaping model version 0.77.
Encoding large.wav to output_large.mp3
Encoding as 44.1 kHz 128 kbps single-ch MPEG1 LayerIII ( 5.5x)  qval=5
    Frame          |  CPU/estimated  |  time/estimated | play/CPU |   ETA
  1149/  1150( 99%)| 0:00:04/ 0:00:04| 0:00:04/ 0:00:04|    7.9615| 0:00:00
[/opt/consumer/lame]$
```

Note: The UNISIM Virtex 5 FXT Simulator must have the hardware FPU enabled (`unisim-virtex5fxt-1.0-wfpu`) to run a Linux kernel for `powerpc-440fp-linux-gnu`. When the simulator has the hardware FPU disabled (`unisim-virtex5fxt-1.0`, the simulator can only use the Linux kernel for `powerpc-440-linux-gnu`.

Appendices

A Simulator technical reference (generated)

This documentation has been automatically generated from the simulator `UNISIM Virtex 5 FXT` version 1.0beta8 on Sep 21 2011.

A.1 Introduction

UNISIM Virtex 5 FXT, full system PPC440x5 based simulator including some Virtex 5 IPs. Section A.2 gives licensing informations about the simulator. Section A.3 shows the set of modules and services that compose the simulator. Section A.4 shows how to invoke the simulator at the command line prompt. Section A.5 gives the simulator parameters. Section A.6 gives the simulator statistic counters. Section A.7 gives the simulator statistic formulas.

A.2 Licensing

UNISIM Virtex 5 FXT 1.0beta8
Copyright (C) 2007-2011, Commissariat a l'Energie Atomique (CEA)
License: BSD (see file COPYING)
Authors: Gilles Mouchard <gilles.mouchard@cea.fr>, Daniel Gracia Pérez <daniel.gracia-perez@cea.fr>

A.3 Simulated configuration

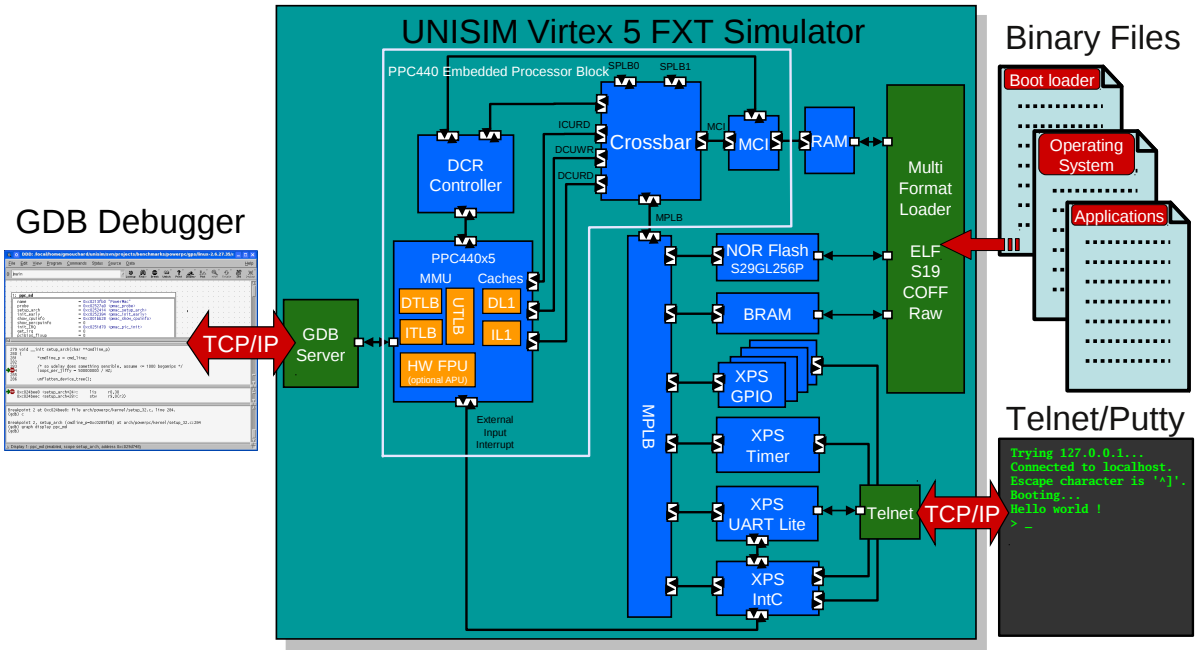

Figure 5: UNISIM Virtex 5 FXT simulator schematic.

The UNISIM Virtex 5 FXT simulator is composed of the following modules and services:

- **5-leds-positions**: This module implements a 5-LED board on GPIO.
- **apu-dcr-stub**: A target stub

- **bram**: this module implements a memory
- **bram-effective-to-physical-address-translator**: this service translates memory addresses when playing with different address sizes
- **capture-trigger-stub0**: A stub that, if enabled, can generate random inputs for a capture timer
- **capture-trigger-stub1**: A stub that, if enabled, can generate random inputs for a capture timer
- **cpu**: This module implements a PPC440 CPU core. It has the following characteristics:
  Processor version (PVR value): 0x7ff21912
  Reset configuration (RSTCFG): U0=0, U1=0, U2=0, U3=0, E=0, ERPN=0x0
  Start address: 0xfffffffc
  L1 data cache: size=32768 bytes, block size=32 bytes, associativity=64
  L1 instruction cache: size=32768 bytes, block size=32 bytes, associativity=64
  shadow instruction TLB: size=4 entries, associativity=4
  shadow data TLB: size=8 entries, associativity=8
  unified TLB: size=64 entries, associativity=64
  FSB/PLB burst size:256 bits
  FSB/PLB width:128 bits
  MMU: yes
  FPU APU: yes

- **critical-input-interrupt-stub**: An initiator stub
- **crossbar**: A crossbar
- **dcr-controller**: A Device Control Register bus controller
- **dip-switches-8bit**: This module implements a 8-switch board (DIP switch or push buttons) on GPIO.

- **dma0-dcr-stub**: A target stub
- **dma1-dcr-stub**: A target stub
- **dma2-dcr-stub**: A target stub
- **dma3-dcr-stub**: A target stub
- **external-slave-dcr-stub**: A target stub
- **flash**: This module implements an S29GL256P flash memory with the following characteristics:
  Manufacturer ID: 0x0001
  Device ID word #0: 0x227e
  Device ID word #1: 0x2222
  Device ID word #2: 0x2201
  Size: 33554432 bytes
  I/O width: 16 bits
  Number of chips: 1 chip
  I/O width per chip: 16 bits
  Size per chip: 33554432 bytes
  Number of Sectors: 256 sectors
  8-bit mode support: yes
  16-bit mode support: yes
  Access time: 100 ns
  Byte programming time: 60000 us
  Word programming time: 60000 us

Sector erasing time: 500000000 us
Chip erasing time: 128000000000 us

- **flash-effective-to-physical-address-translator**: this service translates memory addresses when playing with different address sizes
- **gdb-server**: this service implements the GDB server remote serial protocol over TCP/IP. Standards GDB clients (e.g. gdb, eclipse, ddd) can connect to the simulator to debug the target application that runs within the simulator.
- **generate-out-stub0**: A target stub
- **generate-out-stub1**: A target stub
- **gpio-5-leds-positions**: This module implements a Xilinx XPS GPIO (v2.00a). It has the following characteristics:
  PLB data width: 128 bits
  Use dual channel: no
  GPIO Channel 1 width: 5 bits
  GPIO_DATA reset value: 0x0
  GPIO_TRI reset value: 0xffffffff

- **gpio-dip-switches-8bit**: This module implements a Xilinx XPS GPIO (v2.00a). It has the following characteristics:
  PLB data width: 128 bits
  Use dual channel: no
  GPIO Channel 1 width: 8 bits
  GPIO_DATA reset value: 0x0
  GPIO_TRI reset value: 0xffffffff

- **gpio-leds-8bit**: This module implements a Xilinx XPS GPIO (v2.00a). It has the following characteristics:
  PLB data width: 128 bits
  Use dual channel: no
  GPIO Channel 1 width: 8 bits
  GPIO_DATA reset value: 0x0
  GPIO_TRI reset value: 0xffffffff

- **gpio-push-buttons-5bit**: This module implements a Xilinx XPS GPIO (v2.00a). It has the following characteristics:
  PLB data width: 128 bits
  Use dual channel: no
  GPIO Channel 1 width: 5 bits
  GPIO_DATA reset value: 0x0
  GPIO_TRI reset value: 0xffffffff

- **host-time**: this service is an abstraction layer for the host machine time
- **inline-debugger**: this service implements a built-in debugger in the terminal console
- **input-interrupt-stub0**: An initiator stub
- **input-interrupt-stub1**: An initiator stub
- **input-interrupt-stub10**: An initiator stub
- **input-interrupt-stub11**: An initiator stub
- **input-interrupt-stub12**: An initiator stub

- **input-interrupt-stub13**: An initiator stub
- **input-interrupt-stub14**: An initiator stub
- **input-interrupt-stub15**: An initiator stub
- **input-interrupt-stub16**: An initiator stub
- **input-interrupt-stub17**: An initiator stub
- **input-interrupt-stub18**: An initiator stub
- **input-interrupt-stub19**: An initiator stub
- **input-interrupt-stub20**: An initiator stub
- **input-interrupt-stub21**: An initiator stub
- **input-interrupt-stub22**: An initiator stub
- **input-interrupt-stub23**: An initiator stub
- **input-interrupt-stub24**: An initiator stub
- **input-interrupt-stub25**: An initiator stub
- **input-interrupt-stub26**: An initiator stub
- **input-interrupt-stub27**: An initiator stub
- **input-interrupt-stub28**: An initiator stub
- **input-interrupt-stub29**: An initiator stub
- **input-interrupt-stub30**: An initiator stub
- **input-interrupt-stub31**: An initiator stub
- **input-interrupt-stub4**: An initiator stub
- **input-interrupt-stub5**: An initiator stub
- **input-interrupt-stub6**: An initiator stub
- **input-interrupt-stub9**: An initiator stub
- **intc**: This module implements a Xilinx XPS Interrupt Controller (v2.01a). It has the following characteristics:
  PLB data width: 128 bits
  Number of interrupt inputs: 32 interrupt inputs
  IPR support: yes
  SIE support: yes
  CIE support: yes
  IVR support: yes
  Ouput: active on high level
  input #0 capture mode: rising edge
  input #1 capture mode: rising edge
  input #2 capture mode: rising edge
  input #3 capture mode: rising edge
  input #4 capture mode: rising edge
  input #5 capture mode: rising edge
  input #6 capture mode: rising edge
  input #7 capture mode: rising edge
  input #8 capture mode: rising edge
  input #9 capture mode: rising edge
  input #10 capture mode: rising edge
  input #11 capture mode: rising edge
  input #12 capture mode: rising edge
  input #13 capture mode: rising edge
  input #14 capture mode: rising edge

input #15 capture mode: rising edge
input #16 capture mode: rising edge
input #17 capture mode: rising edge
input #18 capture mode: rising edge
input #19 capture mode: rising edge
input #20 capture mode: rising edge
input #21 capture mode: rising edge
input #22 capture mode: rising edge
input #23 capture mode: rising edge
input #24 capture mode: rising edge
input #25 capture mode: rising edge
input #26 capture mode: rising edge
input #27 capture mode: rising edge
input #28 capture mode: rising edge
input #29 capture mode: rising edge
input #30 capture mode: rising edge
input #31 capture mode: rising edge

- **leds-8bit**: This module implements a 8-LED board on GPIO.

- **loader**: A multi-format loader that supports ELF32, ELF64, S19, COFF and Raw binary files
- **loader.memory-mapper**: A memory mapper
- **loader.tee-backtrace**: This service/client implements a tee ('T'). It unifies the backtrace capability of several services that individually provides their own backtrace capability
- **loader.tee-blob**: This service/client implements a tee ('T'). It unifies the statement lookup capability of several services that individually provides their own statement lookup capability
- **loader.tee-loader**: This service/client implements a tee ('T'). It unifies the loader capability of several services that individually provides their own loader capability
- **loader.tee-stmt-lookup**: This service/client implements a tee ('T'). It unifies the statement lookup capability of several services that individually provides their own statement lookup capability
- **loader.tee-symbol-table-lookup**: This service/client implements a tee ('T'). It unifies the symbol table lookup capability of several services that individually provides their own symbol table lookup capability
- **master1-dcr-stub**: An initiator stub
- **mci**: A Memory Controller Interface (MCI)
- **mplb**: A memory-mapped router
- **push-buttons-5bit**: This module implements a 5-switch board (DIP switch or push buttons) on GPIO.

- **pwm-stub**: A target stub
- **ram**: this module implements a memory
- **ram-effective-to-physical-address-translator**: this service translates memory addresses when playing with different address sizes
- **splb0-stub**: An initiator stub
- **splb1-stub**: An initiator stub

- **time**: this service is an abstraction layer for the SystemC kernel time
- **timer**: This module implements a Xilinx XPS Timer/Counter (v1.02a). It has the following characteristics:
  PLB data width: 128 bits
  Width of the counters: 32 bits
  One timer only: no
  CaptureTrig0 assertion level: high
  CaptureTrig1 assertion level: high
  GenerateOut0 assertion level: high
  GenerateOut0 assertion level: high

- **uart-lite**: This module implements a Xilinx XPS UART Lite (v1.01a). It has the following characteristics:
  PLB data width: 128 bits
  Baud rate: 9600 bits/s
  Data bits: 8 bits
  Parity: odd

A.4 Using the UNISIM Virtex 5 FXT simulator

The UNISIM Virtex 5 FXT simulator has the following command line options:

Usage: `unisim-virtex5fxt-wfpu-1.0beta8 [<options>] [...]`
Options:

- `--set <param=value> or -s <param=value>`: set value of parameter 'param' to 'value'
- `--config <XML file> or -c <XML file>`: configures the simulator with the given XML configuration file
- `--get-config <XML file> or -g <XML file>`: get the simulator configuration XML file (you can use it to create your own configuration. This option can be combined with -c to get a new configuration file with existing variables from another file
- `--list or -l`: lists all available parameters, their type, and their current value
- `--warn or -w`: enable printing of kernel warnings
- `--doc <Latex file> or -d <Latex file>`: enable printing a latex documentation
- `--version or -v`: displays the program version information
- `--share-path <path> or -p <path>`: the path that should be used for the share directory (absolute path)
- `--help or -h`: displays this help

A.5 Configuration

Simulator configuration (see below) can be modified using command line Options `--set <param=value>` or `--config <config file>`.

Global	
Name: `enable-gdb-server`	**Type:** `parameter`
Default: `true`	**Data type:** `boolean`

Valid: `true`, `false`

Description:
Enable/Disable GDB server instantiation.

Name: `enable-inline-debugger` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable inline debugger instantiation.

Name: `enable-press-enter-at-exit` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable pressing key enter at exit.

Name: `enable-telnet` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable telnet instantiation.

Name: `estimate-power` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable power estimators instantiation.

Name: `kernel_logger.file` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Keep logger output in a file.

Name: `kernel_logger.filename` **Type:** `parameter`
Default: `logger_output.txt` **Data type:** `string`

Description:
Filename to keep logger output (the option file must be activated).

Name: `kernel_logger.std_err` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Show logger output through the standard error output.

Name: `kernel_logger.std_err_color` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Colorize logger output through the standard error output (only works if std_err is active).

Name: `kernel_logger.std_out` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Show logger output through the standard output.

Name: `kernel_logger.std_out_color` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Colorize logger output through the standard output (only works if std_out is active).

Name: `kernel_logger.xml_file` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Keep logger output in a file xml formatted.

Name: `kernel_logger.xml_file_gzipped` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
If the xml_file option is active, the output file will be compressed (a .gz extension will be automatically added to the xml_filename option.

Name: `kernel_logger.xml_filename` **Type:** `parameter`
Default: `logger_output.xml` **Data type:** `string`

Description:
Filename to keep logger xml output (the option xml_file must be activated).

5-leds-positions

Name: `5-leds-positions.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

apu-dcr-stub

Name: `apu-dcr-stub.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`

Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `apu-dcr-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

bram

Name: `bram.org` **Type:** `parameter`
Default: `0x00000000fffc0000` **Data type:** `unsigned 64-bit integer`

Description:
memory origin/base address.

Name: `bram.bytesize` **Type:** `parameter`
Default: `262144` **Data type:** `unsigned 64-bit integer`

Description:
memory size in bytes.

Name: `bram.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
memory cycle time.

Name: `bram.read-latency` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
memory read latency.

Name: `bram.write-latency` **Type:** `parameter`
Default: `0 s` **Data type:** `sc_time`

Description:
memory write latency.

Name: `bram.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity.

capture-trigger-stub0

Name: `capture-trigger-stub0.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
cycle time.

Name: `capture-trigger-stub0.nice-time` **Type:** `parameter`
Default: `1 ms` **Data type:** `sc_time`

Description:
nice time.

Name: `capture-trigger-stub0.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `capture-trigger-stub0.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable.

Name: `capture-trigger-stub0.randomized-output` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable randomized output.

Name: `capture-trigger-stub0.random-seed` **Type:** `parameter`
Default: `-123456` **Data type:** `signed 32-bit integer`

Description:
Seed for the random generator.

Name: `capture-trigger-stub0.random-period-min` **Type:** `parameter`
Default: `200` **Data type:** `unsigned 32-bit integer`

Description:
Minimum period in cycles of randomized output.

Name: capture-trigger-stub0.random-↪period-max **Type:** parameter
Default: 799 **Data type:** unsigned 32-bit integer

Description:
Maximum period in cycles of randomized output.

capture-trigger-stub1

Name: capture-trigger-stub1.cycle-↪time **Type:** parameter
Default: 5 ns **Data type:** sc_time

Description:
cycle time.

Name: capture-trigger-stub1.nice-↪time **Type:** parameter
Default: 1 ms **Data type:** sc_time

Description:
nice time.

Name: capture-trigger-stub1.verbose **Type:** parameter
Default: false **Data type:** boolean
Valid: true, false

Description:
Enable/Disable verbosity.

Name: capture-trigger-stub1.enable **Type:** parameter
Default: true **Data type:** boolean
Valid: true, false

Description:
Enable/Disable.

Name: capture-trigger-stub1.randomized-↪output **Type:** parameter
Default: false **Data type:** boolean
Valid: true, false

Description:
Enable/Disable randomized output.

Name: capture-trigger-stub1.random-↪seed **Type:** parameter
Default: -123456 **Data type:** signed 32-bit integer

Description:
Seed for the random generator.

Name: `capture-trigger-stub1.random-↪period-min` **Type:** `parameter`
Default: `200` **Data type:** `unsigned 32-bit integer`

Description:
Minimum period in cycles of randomized output.

Name: `capture-trigger-stub1.random-↪period-max` **Type:** `parameter`
Default: `799` **Data type:** `unsigned 32-bit integer`

Description:
Maximum period in cycles of randomized output.

cpu

Name: `cpu.cpu-cycle-time` **Type:** `parameter`
Default: `2500` **Data type:** `unsigned 64-bit integer`

Description:
CPU cycle time in picoseconds.

Name: `cpu.voltage` **Type:** `parameter`
Default: `1300` **Data type:** `unsigned 64-bit integer`

Description:
CPU voltage in mV.

Name: `cpu.max-inst` **Type:** `parameter`
Default: `18446744073709551615` **Data type:** `unsigned 64-bit integer`

Description:
maximum number of instructions to simulate.

Name: `cpu.verbose-all` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
globally enable/disable verbosity.

Name: `cpu.verbose-setup` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity while setup.

Name: `cpu.verbose-step` **Type:** `parameter`
Default: `false` **Data type:** `boolean`

Valid: `true`, `false`

Description:
enable/disable verbosity when simulating an instruction.

Name: `cpu.verbose-itlb` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when accessing shadow instruction translation lookahead buffer.

Name: `cpu.verbose-dtlb` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when accessing shadow data translation lookahead buffer.

Name: `cpu.verbose-utlb` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when accessing unified translation lookahead buffer.

Name: `cpu.verbose-dl1` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when accessing L1 data cache.

Name: `cpu.verbose-il1` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when accessing L1 instruction cache.

Name: `cpu.verbose-load` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when simulating a load.

Name: `cpu.verbose-store` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when simulating a store.

Name: `cpu.verbose-read-memory` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when reading memory for a debug purpose.

Name: `cpu.verbose-write-memory` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when writing memory for a debug purpose.

Name: `cpu.verbose-exception` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when handling exceptions.

Name: `cpu.verbose-set-msr` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when setting MSR.

Name: `cpu.verbose-tlbwe` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity when executing a tlbwe instruction.

Name: `cpu.enable-linux-printk-snooping` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable linux printk buffer snooping.

Name: `cpu.enable-linux-syscall-snooping` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable linux syscall snooping.

Name: `cpu.trap-on-instruction-counter` **Type:** `parameter`
Default: `18446744073709551615` **Data type:** `unsigned 64-bit integer`

Description:
number of simulated instruction before traping.

Name: `cpu.enable-trap-on-exception` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable trap reporting on exception.

Name: `cpu.bus-cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
bus cycle time.

Name: `cpu.ext-timer-cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
external timer cycle time.

Name: `cpu.nice-time` **Type:** `parameter`
Default: `200 ns` **Data type:** `sc_time`

Description:
maximum time between synchonizations.

Name: `cpu.ipc` **Type:** `parameter`
Default: `1` **Data type:** `double precision floating-point`

Description:
targeted average instructions per second.

Name: `cpu.enable-host-idle` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable host idle periods when target is idle.

critical-input-interrupt-stub

Name: `critical-input-interrupt-stub.↪enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `critical-input-interrupt-stub.↪verbose` **Type:** `parameter`

Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

crossbar

Name: `crossbar.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `crossbar.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Enable/Disable verbosity.

dcr-controller

Name: `dcr-controller.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `dcr-controller.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

dip-switches-8bit

Name: `dip-switches-8bit.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `dip-switches-8bit.polling-↪period` **Type:** `parameter`
Default: `10 ms` **Data type:** `sc_time`

Description:
Polling period of host keyboard.

dma0-dcr-stub

Name: `dma0-dcr-stub.enable` **Type:** `parameter`

Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `dma0-dcr-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

dma1-dcr-stub

Name: `dma1-dcr-stub.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `dma1-dcr-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

dma2-dcr-stub

Name: `dma2-dcr-stub.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `dma2-dcr-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

dma3-dcr-stub

Name: `dma3-dcr-stub.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `dma3-dcr-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

external-slave-dcr-stub

Name: `external-slave-dcr-stub.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `external-slave-dcr-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

flash

Name: `flash.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity.

Name: `flash.org` **Type:** `parameter`
Default: `0x00000000fc000000` **Data type:** `unsigned 64-bit integer`

Description:
flash memory base address.

Name: `flash.bytesize` **Type:** `parameter`
Default: `33554432` **Data type:** `unsigned 64-bit integer`

Description:
flash memory size in bytes.

Name: `flash.endian` **Type:** `parameter`
Default: `little-endian` **Data type:** `endianess`
Valid: `little-endian`, `big-endian`

Description:
endianness of flash memory.

Name: `flash.sector-protect[0]` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable sector write protection.

Name: `flash.sector-protect[1]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[2]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[3]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[4]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[5]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[6]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[7]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[8]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	

Name: `flash.sector-protect[9]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[10]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[11]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[12]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[13]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[14]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[15]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[16]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	

Name: `flash.sector-protect[17]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[18]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[19]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[20]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[21]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[22]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[23]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[24]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[25]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[26]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[27]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[28]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[29]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[30]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[31]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[32]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	

Name: `flash.sector-protect[33]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[34]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[35]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[36]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[37]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[38]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[39]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[40]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[41]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[42]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[43]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[44]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[45]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[46]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[47]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[48]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[49]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[50]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[51]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[52]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[53]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[54]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[55]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[56]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[57]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[58]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[59]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[60]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[61]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[62]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[63]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[64]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	

Name: `flash.sector-protect[65]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[66]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[67]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[68]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[69]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[70]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[71]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[72]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[73]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[74]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[75]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[76]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[77]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[78]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[79]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[80]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[81]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[82]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[83]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[84]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[85]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[86]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[87]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[88]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[89]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[90]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[91]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[92]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[93]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[94]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[95]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[96]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[97]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[98]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[99]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[100]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[101]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[102]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[103]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[104]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[105]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[106]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[107]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[108]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[109]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[110]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[111]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[112]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[113]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[114]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[115]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[116]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[117]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[118]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[119]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[120]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[121]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[122]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[123]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[124]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[125]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[126]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[127]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[128]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[129]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[130]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[131]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[132]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[133]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[134]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[135]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[136]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	

Name: `flash.sector-protect[137]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[138]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[139]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[140]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[141]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[142]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[143]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[144]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	

Name: `flash.sector-protect[145]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[146]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[147]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[148]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[149]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[150]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[151]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[152]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[153]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[154]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[155]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[156]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[157]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[158]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[159]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[160]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[161]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[162]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[163]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[164]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[165]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[166]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[167]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[168]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	

Name: `flash.sector-protect[169]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[170]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[171]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[172]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[173]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[174]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[175]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[176]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[177]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[178]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[179]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[180]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[181]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[182]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[183]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[184]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[185]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[186]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[187]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[188]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[189]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[190]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[191]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[192]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[193]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[194]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[195]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[196]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[197]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[198]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[199]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[200]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[201]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[202]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[203]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[204]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[205]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[206]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[207]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[208]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[209]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[210]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[211]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[212]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[213]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[214]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[215]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[216]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[217]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[218]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[219]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[220]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[221]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[222]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[223]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[224]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[225]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[226]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[227]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[228]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[229]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[230]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[231]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[232]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[233]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[234]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[235]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[236]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[237]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[238]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[239]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	
Name: `flash.sector-protect[240]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description:	
enable/disable sector write protection.	

Name: `flash.sector-protect[241]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[242]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[243]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[244]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[245]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[246]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[247]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[248]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	

Name: `flash.sector-protect[249]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[250]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[251]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[252]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[253]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[254]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.sector-protect[255]`	**Type:** `parameter`
Default: `false`	**Data type:** `boolean`
Valid: `true`, `false`	
Description: enable/disable sector write protection.	
Name: `flash.fsm-to-graphviz-output-↪filename`	**Type:** `parameter`
Default:	**Data type:** `string`

Description:
FSM (finite state machine) to Graphviz output filename.

Name: `flash.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

gdb-server

Name: `gdb-server.memory-atom-size` **Type:** `parameter`
Default: `0x00000001` **Data type:** `unsigned 32-bit integer`

Description:
size of the smallest addressable element in memory.

Name: `gdb-server.tcp-port` **Type:** `parameter`
Default: `0x00000000` **Data type:** `signed 32-bit integer`

Description:
TCP/IP port to listen waiting for a GDB client connection.

Name: `gdb-server.architecture-description-filename` **Type:** `parameter`
Default: `gdb_powerpc.xml` **Data type:** `string`

Description:
filename of a XML description of the connected processor.

Name: `gdb-server.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

generate-out-stub0

Name: `generate-out-stub0.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `generate-out-stub0.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

generate-out-stub1

Name: `generate-out-stub1.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `generate-out-stub1.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

gpio-5-leds-positions

Name: `gpio-5-leds-positions.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `gpio-5-leds-positions.c-baseaddr` **Type:** `parameter`
Default: `0x0000000081420000` **Data type:** `unsigned 64-bit integer`

Description:
Base address (C_BASEADDR design parameter).

Name: `gpio-5-leds-positions.c-highaddr` **Type:** `parameter`
Default: `0x000000008142ffff` **Data type:** `unsigned 64-bit integer`

Description:
High address (C_HIGHADDR design parameter).

Name: `gpio-5-leds-positions.cycle-↪time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

gpio-dip-switches-8bit

Name: `gpio-dip-switches-8bit.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `gpio-dip-switches-8bit.c-baseaddr` **Type:** `parameter`

Default: `0x0000000081460000` **Data type:** `unsigned 64-bit integer`

Description:
Base address (C_BASEADDR design parameter).

Name: `gpio-dip-switches-8bit.c-highaddr` **Type:** `parameter`
Default: `0x0000000081460000` **Data type:** `unsigned 64-bit integer`

Description:
High address (C_HIGHADDR design parameter).

Name: `gpio-dip-switches-8bit.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

gpio-leds-8bit

Name: `gpio-leds-8bit.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `gpio-leds-8bit.c-baseaddr` **Type:** `parameter`
Default: `0x0000000081400000` **Data type:** `unsigned 64-bit integer`

Description:
Base address (C_BASEADDR design parameter).

Name: `gpio-leds-8bit.c-highaddr` **Type:** `parameter`
Default: `0x0000000081460000` **Data type:** `unsigned 64-bit integer`

Description:
High address (C_HIGHADDR design parameter).

Name: `gpio-leds-8bit.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

gpio-push-buttons-5bit

Name: `gpio-push-buttons-5bit.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `gpio-push-buttons-5bit.c-baseaddr` **Type:** `parameter`
Default: `0x0000000081440000` **Data type:** `unsigned 64-bit integer`

Description:
Base address (C_BASEADDR design parameter).

Name: `gpio-push-buttons-5bit.c-highaddr` **Type:** `parameter`
Default: `0x000000008144ffff` **Data type:** `unsigned 64-bit integer`

Description:
High address (C_HIGHADDR design parameter).

Name: `gpio-push-buttons-5bit.cycle-↪time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

inline-debugger

Name: `inline-debugger.memory-atom-↪size` **Type:** `parameter`
Default: `0x00000001` **Data type:** `unsigned 32-bit integer`

Description:
size of the smallest addressable element in memory.

Name: `inline-debugger.num-loaders` **Type:** `parameter`
Default: `4` **Data type:** `unsigned 32-bit integer`

Description:
number of loaders.

Name: `inline-debugger.search-path` **Type:** `parameter`
Default: **Data type:** `string`

Description:
Search path for source (separated by ';').

Name: `inline-debugger.init-macro` **Type:** `parameter`
Default: **Data type:** `string`

Description:
path to initial macro to run when debugger starts.

Name: `inline-debugger.output` **Type:** `parameter`
Default: **Data type:** `string`

Description:
path to output file where to redirect the debugger outputs.

input-interrupt-stub0

Name: `input-interrupt-stub0.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub0.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub1

Name: `input-interrupt-stub1.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub1.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub10

Name: `input-interrupt-stub10.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub10.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub11

Name: `input-interrupt-stub11.enable` **Type:** `parameter`

Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub11.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub12

Name: `input-interrupt-stub12.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub12.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub13

Name: `input-interrupt-stub13.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub13.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub14

Name: `input-interrupt-stub14.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub14.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub15

Name: `input-interrupt-stub15.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub15.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub16

Name: `input-interrupt-stub16.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub16.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub17

Name: `input-interrupt-stub17.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub17.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub18

Name: `input-interrupt-stub18.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub18.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub19

Name: `input-interrupt-stub19.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub19.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub20

Name: `input-interrupt-stub20.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub20.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub21

Name: `input-interrupt-stub21.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub21.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub22

Name: `input-interrupt-stub22.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub22.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub23

Name: `input-interrupt-stub23.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub23.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub24

Name: `input-interrupt-stub24.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub24.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub25

Name: `input-interrupt-stub25.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub25.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub26

Name: `input-interrupt-stub26.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub26.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub27

Name: `input-interrupt-stub27.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub27.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub28

Name: `input-interrupt-stub28.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub28.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub29

Name: `input-interrupt-stub29.enable` **Type:** `parameter`

Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub29.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub30

Name: `input-interrupt-stub30.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub30.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub31

Name: `input-interrupt-stub31.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub31.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub4

Name: `input-interrupt-stub4.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub4.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub5

Name: `input-interrupt-stub5.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub5.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub6

Name: `input-interrupt-stub6.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub6.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

input-interrupt-stub9

Name: `input-interrupt-stub9.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `input-interrupt-stub9.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

intc

Name: `intc.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `intc.c-baseaddr` **Type:** `parameter`
Default: `0x0000000081800000` **Data type:** `unsigned 64-bit integer`

Description:
Base address (C_BASEADDR design parameter).

Name: `intc.c-highaddr` **Type:** `parameter`
Default: `0x000000008180ffff` **Data type:** `unsigned 64-bit integer`

Description:
High address (C_HIGHADDR design parameter).

Name: `intc.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

leds-8bit

Name: `leds-8bit.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

loader

Name: `loader.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `loader.verbose-parser` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity of parser.

Name: `loader.filename` **Type:** `parameter`
Default: **Data type:** `string`

Description:
List of files to load. Syntax: [[filename=]<filename1>[:[format=]<format1>]][,[filename=]<filename2>[:[form
(e.g. boot.bin:raw,app.elf).

loader.memory-mapper

Name: `loader.memory-mapper.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `loader.memory-mapper.verbose-parser` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity of parser.

Name: `loader.memory-mapper.mapping` **Type:** `parameter`
Default: `ram-effective-to-physical-address-translator:0x00000000-0x0fffffff,bram-effective-to-physical-address-translator:0xfffc0000-0xffffffff,flash-effective-to-physical-address-translator:0xfc000000-0xfdffffff` **Data type:** `string`

Description:
Memory mapping. Syntax: [[(memory=]<memory1>[:[range=]<low1-high1>]][,[(memory=]<memory2>[:[range=]<low2-high2>]]... (e.g. ram:0x0-0x00ffff,rom:0xff0000-0xffffff).

master1-dcr-stub

Name: `master1-dcr-stub.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `master1-dcr-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

mci

Name: `mci.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: mci.cycle-time
Type: parameter
Default: 5 ns
Data type: sc_time

Description:
Enable/Disable verbosity.

mplb

Name: mplb.cycle_time
Type: parameter
Default: 5 ns
Data type: sc_time

Description:
Time to process a request/response by the router.

Name: mplb.port_buffer_size
Type: parameter
Default: 0x00000000
Data type: unsigned 32-bit integer

Description:
Defines the size of the buffer for incomming requests in each of the input ports (0 = infinite).

Name: mplb.mapping_0
Type: parameter
Default: range_start="0x81800000" ↪range_end="0x8180ffff" output_ ↪port="0" translation="0x81800000"
Data type: unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

Name: mplb.mapping_1
Type: parameter
Default: range_start="0x83c00000" ↪range_end="0x83c0ffff" output_ ↪port="1" translation="0x83c00000"
Data type: unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

Name: mplb.mapping_2
Type: parameter
Default: range_start="0xfc000000" ↪range_end="0xfdffffff" output_ ↪port="2" translation="0xfc000000"
Data type: unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

Name: mplb.mapping_3
Type: parameter
Default: range_start="0xfffc0000" ↪range_end="0xffffffff" output_ ↪port="3" translation="0xfffc0000"
Data type: unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

Name: mplb.mapping_4
Type: parameter
Default: range_start="0x84000000" ↪range_end="0x8400ffff" output_ ↪port="4" translation="0x84000000"
Data type: unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

Name: mplb.mapping_5
Type: parameter
Default: range_start="0x81460000" ↪range_end="0x8146ffff" output_ ↪port="5" translation="0x81460000"
Data type: unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

Name: mplb.mapping_6
Type: parameter
Default: range_start="0x81400000" ↪range_end="0x8140ffff" output_ ↪port="6" translation="0x81400000"
Data type: unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

Name: mplb.mapping_7 **Type:** parameter
Default: range_start="0x81420000" ↪range_end="0x8142ffff" output_ ↪port="7" translation="0x81420000" **Data type:** unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

Name: mplb.mapping_8 **Type:** parameter
Default: range_start="0x81440000" ↪range_end="0x8144ffff" output_ ↪port="8" translation="0x81440000" **Data type:** unisim::component::tlm2::interconnect: ↪:generic_router::Mapping

Description:
Defined a mapping of the router with format "[range_start]","[range_end]","[outport_index]" where [range_start], [range_end] and [outport_index] are to be replaced with the initial address, end address (= range_start + range_size - 1) and the output port index respectively.

push-buttons-5bit

Name: push-buttons-5bit.verbose **Type:** parameter
Default: false **Data type:** boolean
Valid: true, false

Description:
Enable/Disable verbosity.

Name: push-buttons-5bit.polling- ↪period **Type:** parameter
Default: 10 ms **Data type:** sc_time

Description:
Polling period of host keyboard.

pwm-stub

Name: pwm-stub.enable **Type:** parameter
Default: true **Data type:** boolean
Valid: true, false

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: pwm-stub.verbose **Type:** parameter
Default: false **Data type:** boolean
Valid: true, false

Description:
Enable/Disable verbosity.

ram

Name: `ram.org` **Type:** `parameter`
Default: `0x0000000000000000` **Data type:** `unsigned 64-bit integer`

Description:
memory origin/base address.

Name: `ram.bytesize` **Type:** `parameter`
Default: `268435456` **Data type:** `unsigned 64-bit integer`

Description:
memory size in bytes.

Name: `ram.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
memory cycle time.

Name: `ram.read-latency` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
memory read latency.

Name: `ram.write-latency` **Type:** `parameter`
Default: `0 s` **Data type:** `sc_time`

Description:
memory write latency.

Name: `ram.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
enable/disable verbosity.

splb0-stub

Name: `splb0-stub.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `splb0-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

splb1-stub

Name: `splb1-stub.enable` **Type:** `parameter`
Default: `true` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable a lazy implementation of TLM 2.0 method interface.

Name: `splb1-stub.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

timer

Name: `timer.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`
Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `timer.c-baseaddr` **Type:** `parameter`
Default: `0x0000000083c00000` **Data type:** `unsigned 64-bit integer`

Description:
Base address (C_BASEADDR design parameter).

Name: `timer.c-highaddr` **Type:** `parameter`
Default: `0x0000000083c0ffff` **Data type:** `unsigned 64-bit integer`

Description:
High address (C_HIGHADDR design parameter).

Name: `timer.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

uart-lite

Name: `uart-lite.verbose` **Type:** `parameter`
Default: `false` **Data type:** `boolean`

Valid: `true`, `false`

Description:
Enable/Disable verbosity.

Name: `uart-lite.c-baseaddr` **Type:** `parameter`
Default: `0x0000000084000000` **Data type:** `unsigned 64-bit integer`

Description:
Base address (C_BASEADDR design parameter).

Name: `uart-lite.c-highaddr` **Type:** `parameter`
Default: `0x000000008400ffff` **Data type:** `unsigned 64-bit integer`

Description:
High address (C_HIGHADDR design parameter).

Name: `uart-lite.cycle-time` **Type:** `parameter`
Default: `5 ns` **Data type:** `sc_time`

Description:
Cycle time.

Name: `uart-lite.telnet-refresh-time` **Type:** `parameter`
Default: `10 ms` **Data type:** `sc_time`

Description:
Telnet refresh time.

A.6 Statistics

Simulation statistic counters are listed below:

bram

Name: `bram.memory-usage` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
host memory usage in bytes of simulated memory.

cpu

Name: `cpu.instruction-counter` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of simulated instructions.

Name: `cpu.timer-cycle` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of simulated timer cycles.

Name: `cpu.num-il1-accesses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of accesses to L1 instruction cache.

Name: `cpu.num-il1-misses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of misses to L1 instruction cache.

Name: `cpu.num-dl1-accesses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of accesses to L1 data cache.

Name: `cpu.num-dl1-misses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of misses to L1 data cache.

Name: `cpu.num-itlb-accesses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of accesses to shadow instruction translation look-aside buffer.

Name: `cpu.num-itlb-misses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of misses to shadow instruction translation look-aside buffer.

Name: `cpu.num-dtlb-accesses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of accesses to shadow data translation look-aside buffer.

Name: `cpu.num-dtlb-misses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of misses to shadow data translation look-aside buffer.

Name: `cpu.num-utlb-accesses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of accesses to unified data translation look-aside buffer.

Name: `cpu.num-utlb-misses` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
number of misses to unified data translation look-aside buffer.

Name: `cpu.num-interrupts` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
Number of interrupts.

Name: `cpu.run-time` **Type:** `statistic`
Data type: `sc_time`

Description:
run time.

Name: `cpu.idle-time` **Type:** `statistic`
Data type: `sc_time`

Description:
idle time.

ram

Name: `ram.memory-usage` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
host memory usage in bytes of simulated memory.

timer

Name: `timer.num-tcr0-roll-over` **Type:** `statistic`
Data type: `unsigned 64-bit integer`

Description:
Number of timer/counter 0 roll over.

Name: `timer.num-tcr1-roll-over`	**Type:** `statistic` **Data type:** `unsigned 64-bit integer`
Description: Number of timer/counter 1 roll over.	
Name: `timer.num-timer0-generate-↪interrupts`	**Type:** `statistic` **Data type:** `unsigned 64-bit integer`
Description: Number of timer 0 generate interrupts.	
Name: `timer.num-timer1-generate-↪interrupts`	**Type:** `statistic` **Data type:** `unsigned 64-bit integer`
Description: Number of timer 1 generate interrupts.	
Name: `timer.num-timer0-generate-↪interrupt-losses`	**Type:** `statistic` **Data type:** `unsigned 64-bit integer`
Description: Number of timer 0 generate interrupt losses.	
Name: `timer.num-timer1-generate-↪interrupt-losses`	**Type:** `statistic` **Data type:** `unsigned 64-bit integer`
Description: Number of timer 1 generate interrupt losses.	
Name: `timer.num-timer0-captures`	**Type:** `statistic` **Data type:** `unsigned 64-bit integer`
Description: Number of timer 0 captures.	
Name: `timer.num-timer1-captures`	**Type:** `statistic` **Data type:** `unsigned 64-bit integer`
Description: Number of timer 1 captures.	
Name: `timer.num-timer0-old-capture-↪losses`	**Type:** `statistic`

	Data type: `unsigned 64-bit integer`
Description: Number of timer 0 old capture losses.	
Name: `timer.num-timer1-old-capture-losses`	**Type:** `statistic`
	Data type: `unsigned 64-bit integer`
Description: Number of timer 1 old capture losses.	
Name: `timer.num-timer0-new-capture-losses`	**Type:** `statistic`
	Data type: `unsigned 64-bit integer`
Description: Number of timer 0 new capture losses.	
Name: `timer.num-timer1-new-capture-losses`	**Type:** `statistic`
	Data type: `unsigned 64-bit integer`
Description: Number of timer 1 new capture losses.	

A.7 Formulas

Simulation statistic formulas are listed below:

cpu	
Name: `cpu.il1-miss-rate`	**Type:** `formula`
Formula: `cpu.num-il1-misses / cpu.num-il1-accesses`	**Data type:** `double precision floating-point`
Description: L1 instruction cache miss rate.	
Name: `cpu.dl1-miss-rate`	**Type:** `formula`
Formula: `cpu.num-dl1-misses / cpu.num-dl1-accesses`	**Data type:** `double precision floating-point`
Description: L1 data cache miss rate.	
Name: `cpu.itlb-miss-rate`	**Type:** `formula`
Formula: `cpu.num-itlb-misses / cpu.num-itlb-accesses`	**Data type:** `double precision floating-point`

Description:
shadow instruction translation look-aside buffer miss rate.

Name: cpu.dtlb-miss-rate **Type:** formula
Formula: cpu.num-dtlb-misses / cpu.num-dtlb-accesses **Data type:** double precision floating-point

Description:
shadow data translation look-aside buffer miss rate.

Name: cpu.utlb-miss-rate **Type:** formula
Formula: cpu.num-utlb-misses / cpu.num-utlb-accesses **Data type:** double precision floating-point

Description:
unified data translation look-aside buffer miss rate.

Name: cpu.idle-rate **Type:** formula
Formula: cpu.idle-time / cpu.run-time **Data type:** double precision floating-point

Description:
idle rate.

Name: cpu.load-rate **Type:** formula
Formula: 1 - cpu.idle-time / cpu.run-time **Data type:** double precision floating-point

Description:
load rate.

timer

Name: timer.num-timer0-capture-losses **Type:** formula
Formula: timer.num-timer0-old-capture-losses + timer.num-timer0-new-capture-losses **Data type:** signed 64-bit integer

Description:
Number of timer 0 capture losses.

Name: timer.num-timer1-capture-losses **Type:** formula
Formula: timer.num-timer1-old-capture-losses + timer.num-timer1-new-capture-losses **Data type:** signed 64-bit integer

Description:
Number of timer 1 capture losses.

References

[1] MiBench Version 1.0. http://www.eecs.umich.edu/mibench.

[2] S-Record Output Format. 1992. In MC68000 Family Programmers Reference Manual, Appendix C, http://www.freescale.com/files/archives/doc/ref_manual/M68000PRM.pdf.

[3] ELF-64 Object File Format. May 1998. http://downloads.openwatcom.org/ftp/devel/docs/elf-64-gen.pdf.

[4] PPC440x5 CPU Core User's Manual. IBM, July 2003. http://www-01.ibm.com/chips/techlib/techlib.nsf/techdocs/622DCFA1D98B1F10002575A7005464AF/$file/ppc440x5_um.pdf.

[5] DWARF Debugging Information Format Version 3. Free Standards Group, December 2005. http://www.dwarfstd.org/doc/Dwarf3.pdf.

[6] S29GL-P MirrorBit^{TM} Flash Family. SPANSION, November 2006. http://www.spansion.com/Support/Datasheets/S29GL-P_00_A13_e.pdf.

[7] XPS General Purpose Input/Output (GPIO) (v2.00a). Xilinx, December 2009. DS569, http://www.xilinx.com/support/documentation/ip_documentation/xps_gpio.pdf.

[8] XPS UART Lite (v1.01a). Xilinx, December 2009. DS571, http://www.xilinx.com/support/documentation/ip_documentation/xps_uartlite.pdf.

[9] Embedded Processor Block in Virtex-5 FPGAs (v1.8) Reference Guide. Xilinx, February 2010. UG200, http://www.xilinx.com/support/documentation/user_guides/ug200.pdf.

[10] LogiCORE IP XPS Interrupt Controller (v2.01a). Xilinx, April 2010. DS572, http://www.xilinx.com/support/documentation/ip_documentation/xps_intc.pdf.

[11] LogiCORE IP XPS Timer/Counter (v1.02a). Xilinx, April 2010. DS573, http://www.xilinx.com/support/documentation/ip_documentation/xps_timer.pdf.

[12] LogiCORE IP Virtex-5 APU Floating-Point Unit (v1.01a). Xilinx, March 2011. DS693, http://www.xilinx.com/support/documentation/ip_documentation/apu_fpu_virtex5.pdf.

[13] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown. Mibench: A free, commercially representative embedded benchmark suite. In Proceedings of the Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop, pages 3–14, Washington, DC, USA, 2001. IEEE Computer Society.

[14] Richard Stallman, Roland Pesch, Stan Shebs, and al. Debugging with GDB. GNU. http://sourceware.org/gdb/current/onlinedocs/gdb.pdf.gz.